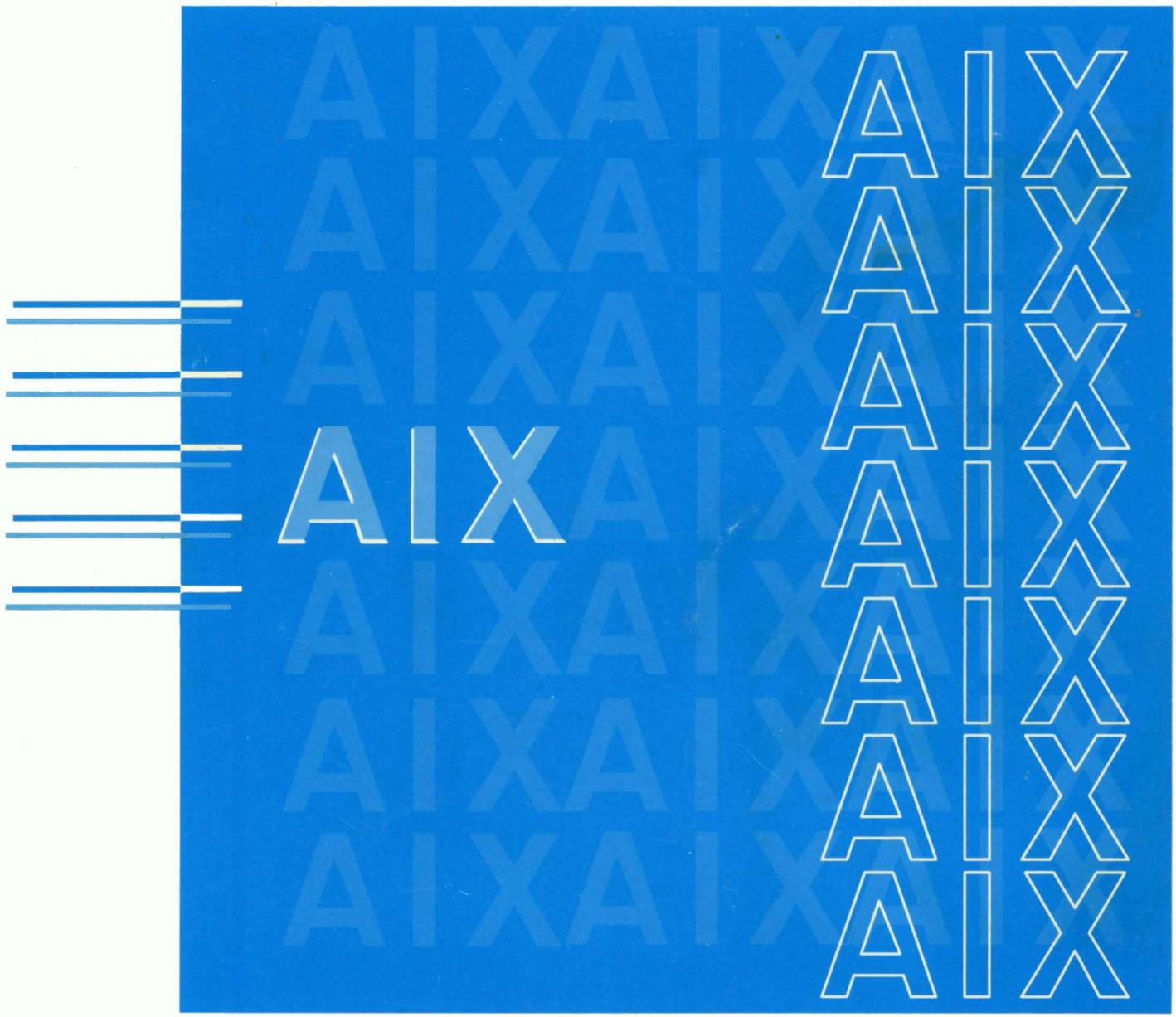
AIX Access for DOS User's

User's Guide



IBM

First Edition (March 1989)

This edition applies to Version 1.1 of the IBM AIX Access for DOS User's Licensed Program, and to all subsequent releases until otherwise indicated in new editions or technical newsletters. Changes are made periodically to the information herein; these changes will be reported in technical newsletters or in new editions of this publication.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

International Business Machines Corporation provides this manual "as is," without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this manual at any time.

Products are not stocked at the address given below. Requests for copies of this product and for technical information about the system should be made to your IBM authorized RT dealer, your IBM marketing representative, or your IBM authorized remarketer.

Address any comments you have to IBM Corporation, Department 997, 11400 Burnet Road, Austin, Texas 78758-3493. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1989
© Copyright Locus Computing Corporation 1986, 1988

Trademarks

The following trademarks apply to this book.

- IBM is a registered trademark of International Business Machines Corporation.
- Personal Computer AT and AT are registered trademarks of International Business Machines Corporation.
- Personal Computer XT and PC XT are registered trademarks of International Business Machines Corporation.
- RT is a registered trademark of International Business Machines Corporation.
- AIX is a trademark of International Business Machines Corporation.
- PS/2 is a registered trademark of International Business Machines Corporation.
- 3Com EtherLink is a trademark of 3Com Corporation.
- SmartCom II is a trademark of Hayes Microcomputer Products, Inc.
- Lotus and 1-2-3 are trademarks of Lotus Development Corporation.
- DEC and VT100 are trademarks of Digital Equipment Corporation.
- XEROX is a trademark of XEROX, Inc.
- Micro Channel is a trademark of International Business Machines Corporation.
- Lattice is a trademark of Lattice, Inc.
- XENIX is a trademark of Microsoft Corporation.
- dBase III Plus is a trademark of the Ashton-Tate Corporation.
- Multimate is a trademark of Multimate International Corporation, an Ashton-Tate company.
- Sidekick is a trademark of Borland International, Inc.
- Turbo Pascal is a trademark of Borland International, Inc.
- Turbo Prolog is a trademark of Borland International, Inc.
- GEM is a trademark of Digital Research, Inc.
- GEM Draw Plus is a trademark of Digital Research, Inc.
- Sideways is a trademark of Funk Software, Inc.
- DisplayWrite 4 is a trademark of International Business Machines Corporation.
- Wordstar is a trademark of MicroPro International Corporation.
- MS is a trademark of Microsoft Corporation.

About This Book

This book explains how to use AIX Access for DOS Users (the Access program). It describes the Access program software, how to install the program software, how to use the software once it is installed, and how to use the Access program commands.

Who Should Use This Book

This book is for users who have a general knowledge of IBM DOS Version 3.3, and who are interested in taking advantage of information or services that exist in an AIX host computer.

Experienced DOS and AIX users who want more information about the structure of the Access program and users responsible for installing and administering the Access program software on the AIX host should refer to *AIX Access for DOS Users Administrator's Guide.*

How to Use This Book

This book is divided into seven chapters and three appendixes, each devoted to a major Access program concept or feature.

- Chapter 1, "Introduction," introduces the Access program software and the configuration requirements for your personal computer.
- Chapter 2, "Using Host File Services," explains how to establish communications with your AIX host and how to run DOS applications using the host as a large fixed disk.
- Chapter 3, "The on Utilities," describes how to execute AIX commands from your personal computer running DOS.
- Chapter 4, "Using Your Computer as an AIX Terminal," describes the terminal emulation features of the Access program.
- Chapter 5, "Tailoring AIX Access for DOS Users," contains tips and ideas for modifying the Access program installation to suit your needs.
- Chapter 6, "The AIX Access for DOS Users vi Editor," describes how to use the DOS version of the AIX **vi** command.
- Chapter 7, "Command Summary," provides a quick reference for all the Access program commands discussed in this book.
- Appendix A, "Installing AIX Access for DOS Users," is in two parts. Part one provides instructions for installing an asynchronous communications adapter or a network interface adapter in your personal computer. Part two describes combining the Access program software with DOS to create a working version of the Access program.
- Appendix B, "Messages," explains the messages you might receive from the Access program.
- Appendix C, "Using a Dial-Up Modem," explains how to establish a host connection using a dial-up modem.
- Appendix D, "Extended Library - PCILIB," discusses the PCILIB programming library that allows DOS programs to use the extended I/O control functions of the Access program.

Throughout this book, the term ***personal computer*** refers to the IBM Personal Computer, the IBM Personal Computer XT, the IBM Personal Computer AT, the IBM Personal Computer XT Model 286 or the IBM PS/2.

The term ***DOS*** refers to the IBM Disk Operating System program (Version 3.3).

The term ***AIX*** refers to any one of AIX Operating System family running AIX DOS Server.

The term ***LAN*** (local area network) refers to a connection path from your personal computer to a host via an Ethernet or Token-Ring communication protocol.

Related Publications

AIX Access for DOS Users Administrator's Guide explains how to install and administer the AIX Access for DOS Users Program.

AIX Operating System Technical Reference describes the files, system calls, and devices for the AIX Operating System.

AIX Operating System Commands Reference lists and describes the AIX Operating System commands.

Contents

Chapter 1. Introduction **1-1**
About This Chapter 1-3
AIX Access for DOS Users 1-4
Minimum Requirements 1-4
Installation 1-5

Chapter 2. Using Host File Services **2-1**
About This Chapter 2-3
Beginning a File Services Session 2-4
Logging in over a LAN Connection Path 2-5
Logging in over an RS-232 Connection Path 2-6
Logging in to Additional Hosts 2-7
Logging in from the Command Line 2-8
Using the AIX File System 2-9
AIX and DOS File Names 2-9
Using AIX and DOS Text Files 2-13
AIX File Protection 2-14
AIX and DOS Commands 2-17
Printing with AIX Access for DOS Users 2-17
Ending a File Services Session 2-20

Chapter 3. The on Utilities **3-1**
About This Chapter 3-3
Uses of the on Utilities 3-4
Using the on Command 3-4
The on Command and .exe Files 3-5
Linking the on Command to .exe Files 3-6
Setting Environment Variables 3-7
Expanding DOS Environment Space 3-8
Breaking Out of on 3-8
Job Control 3-9
The Job Table 3-9
Detaching Tasks 3-9
Keeping Track of Detached Tasks 3-10
Reattaching to a Detached Process 3-11
Saving Output from Completed Jobs 3-12
Stopping Detached Jobs 3-12
Pipes and Redirection 3-13
DOS Pipes 3-13
DOS Output Redirection 3-13
DOS Input Redirection 3-13
Using Pipes and Redirection in Detached Tasks 3-14
Summary of Restrictions and Cautions 3-15

Chapter 4. Using Your Computer as an AIX Terminal **4-1**
About This Chapter 4-3
The Terminal Emulation Utility 4-4
Beginning a Terminal Emulation Session 4-4
Communications Parameters 4-5
Logging in over a LAN Connection Path 4-7
Logging in over an RS-232 Connection Path 4-8
Switching between Terminal Emulation and DOS 4-9
Closing a Terminal Emulation Session 4-10
Using the Help Menu 4-11
The Personal Computer Keyboard 4-12
The Standard Typewriter Area 4-12

The Numeric Keypad 4-13
The Function Keys 4-13
PC Scancode Key Combinations 4-14
Send and Receive Files 4-15
Using a Receive File 4-15
Using a Send File 4-16

Chapter 5. Tailoring AIX Access for DOS Users 5-1
About This Chapter 5-3
AIX Access for DOS Users Program Files 5-4
Putting AIX Access for DOS Users in a Subdirectory 5-5
Putting AIX Access for DOS Users Files on a Virtual Drive 5-6
Putting DOS Applications on a Virtual Drive 5-6
Customizing File Services 5-7
Setting Search Paths 5-7
Setting a Default Search Path 5-8
Changing the Maximum Number of Virtual Drives 5-8
Changing the Maximum Number of Open Files under DOS 5-8
Changing the Job Table Size 5-9
Setting the Printer Environment 5-9
Setting the on Command Variables 5-9
The ONNAME Environment Variable 5-10
The ONPREFIX Environment Variable 5-10
Setting the Terminal Emulation Environment Variable 5-11
Using Terminal Emulation to an AIX/RT Host 5-11

Chapter 6. The AIX Access for DOS Users vi Editor 6-1
About This Chapter 6-3
Invoking the AIX Access for DOS Users vi Editor 6-4
Keyboard Features 6-4
Working Files 6-5
Configuration Options 6-5
File Recovery 6-6
Summary of Differences 6-6

Chapter 7. Command Summary 7-1
About This Chapter 7-3
Syntax Diagrams 7-4
Command, Flag, and Parameter Notation 7-6
aix2dos 7-7
doswhat 7-9
dos2aix 7-10
em 7-11
exrecovr 7-12
jobs 7-13
kill 7-14
login 7-16
logout 7-18
nty 7-19
on 7-20
pciinit 7-24
printer 7-25
setdebug Command 7-28
udir 7-32
vi 7-33

Appendix A. Installing AIX Access for DOS Users A-1
About This Appendix A-3
Installing the Asynchronous Communication Adapter A-4
Installing the Network Interface Adapter A-4
3Com EtherLink Adapter A-4
Ungermann-Bass Network Interface Adapter A-4

The IBM Token-Ring Network PC Adapter A-5
Package Types A-6
Overview of Installing AIX Access for DOS Users A-7
Two Diskette-Drive System A-8
Making Backup Diskettes on a Two Diskette-Drive System A-8
Installing AIX Access for DOS Users on a Two Diskette-Drive System A-10
One Diskette-Drive System A-13
Making Backup Diskettes on a One Diskette-Drive System A-13
Installing AIX Access for DOS Users on a One Diskette-Drive System Using a RAM Disk A-14
Fixed-Disk Drive A-18
Making Backup Diskettes for a Fixed-Disk Drive A-18
Installing AIX Access for DOS Users on a Fixed-Disk Drive A-20

Appendix B. Messages B-1
PCI-vi Error Messages B-8

Appendix C. Using a Dial-Up Modem C-1
Establishing a Host File Services Session C-1
Establishing a Terminal Emulation Session C-2

Appendix D. Extended Library - PCILIB D-1
About This Appendix D-3
What Is PCILIB? D-4
Library Formats D-4
Memory Models Supported D-4
Include Files D-5
Manifest Constants D-5
Data Structures D-6
dflthost D-9
getuattr D-10
isvirtual D-11
mapd2u D-12
mapu2d D-13
msgctl D-14
msgget D-16
msgop D-18
semctl D-21
semget D-23
semop D-25
uchmod D-27
uexec D-28
ukill D-30
uren D-32
uwait D-33
vdrive D-34
vhost D-35

Index X-1

Chapter 1. Introduction

CONTENTS

About This Chapter 1-3
AIX Access for DOS Users 1-4
Minimum Requirements 1-4
Installation 1-5
Adapter 1-5
Software 1-5

About This Chapter

This chapter introduces the Access program software. It lists the minimum system requirements for running the software, and describes the installation process. In addition, this chapter provides a list of the compatible DOS software products that operate with the Access program.

AIX Access for DOS Users

The Access program software package provides a means for exchanging files, and accessing applications between a personal computer and the AIX family of operating systems. The Access program takes advantage of the storage capacity of the AIX file system, allowing personal computers without fixed disks to operate as efficiently as those with fixed disks.

Using the Access program with a LAN or asynchronous connection to an AIX host, you can do the following:

- Run DOS applications using data from the AIX file system
- Store files and DOS applications on the AIX host and access them as if they were on a local fixed disk
- Use the security capabilities of AIX to share files throughout a personal computer network while protecting your files from unauthorized access
- Use host resources, such as printers
- Emulate an ASCII terminal with your personal computer, allowing it to conduct an AIX session on the host
- Emulate a PC Console (scancode terminal) with your personal computer,[1] allowing it to work with DOS applications running on the AIX host.

DOS users do not need to be familiar with AIX to use host file services. The AIX system performs like an extra disk drive connected directly to a personal computer. Users can combine host file services and terminal emulation, and toggle back and forth between the two modes. For example, you could create a text file in host file services mode using a DOS word-processing package, then switch to terminal emulation mode to include that file in an AIX mail message.

Minimum Requirements

To use the Access program software, you must have the following:

- An IBM Personal Computer, IBM Personal Computer XT, IBM Personal Computer XT Model 286, IBM Personal Computer AT, or IBM PS/2 with the following features:
  - At least 256KB of memory
  - At least one 3.5-inch or 5.25-inch diskette drive
  - IBM DOS Version 3.3
  - A LAN connection or an asynchronous communications adapter as described in Chapter One of *AIX Access for DOS Users Administrator's Guide.*
- A host with the following features:
  - The AIX Operating System
  - The AIX DOS Server software
  - A LAN or RS-232 adapter.

---

1 PC Scancode terminal emulation is not supported with an IBM RT host computer.

Installation

Chapter 2, "Using Host File Services," through Chapter 4, "Using Your Computer as an AIX Terminal," of this book assume you have installed a LAN or RS-232 connector and the Access program as described in Appendix A, "Installing AIX Access for DOS Users."

Adapter

The types of adapters you install in your personal computer depends on the types of adapters installed in the AIX hosts that you will attach to. You can install more than one LAN adapter (Ethernet or Token-Ring) in your personal computer, but you can only use one at a time. You can, however, establish more than one AIX host session with any LAN adapter. You can also establish an AIX host session over an RS-232 connection path at the same time you are using a LAN connection.

Software

To use the Access program, you must first install it either by creating working diskettes, or by installing the Access program on your fixed-disk drive so that it is automatically initialized when you start your personal computer.

The Access program distribution diskettes contain an installation program called **install** that simplifies the installation process. It allows you to install the basic system, the standard system, or the full development system. Before reading further, you should follow the instructions in Appendix A, "Installing AIX Access for DOS Users."

Copy Protection

The Access program software is copy-protected. Although you can make copies for your personal use, you cannot use two copies at the same time. When the AIX host detects two simultaneous sessions from the same distribution diskette, it disables the session of the user who logged in last. This can happen any time after the second user has logged in and begun working.

Compatibility

The following DOS application software products operate with the Access program:

- Multimate Advantage II
- dBase III Plus
- The Print Shop
- The Print Shop Graphics Library
- Sidekick, Version 1.5
- Turbo Pascal, Version 3
- Turbo Pascal, Version 4
- Turbo Prolog
- Turbo Graphix Toolbox
- GEM Draw Plus
- GEM Collection
- Sideways
- DisplayWrite 4
- BASICA, Version 3.3

- Lotus 1-2-3, Version 2.01
- WordStar Professional
- MS Windows, Version 1.03
- MS MASM, Version 5.0
- MS Quick C, Version 5.0
- MS Quick BASIC
- MS Flight Simulator
- Webster's Spelling Check
- Ventura Publisher, Release 1.1.

Other DOS application software products can also operate with the Access program provided they do not interfere with the normal functions of the system configuration. Interference occurs when programs do any of the following:

- Overlay the DOS or BIOS area of storage
- Program the 8259 interrupt controller in a way that interferes with the Access program's use of that controller
- Disable interrupts, fail to issue an end-of-interrupt or IRET on a hardware interrupt level, or mask selected interrupt levels for more than 100 milliseconds
- Use interrupts 13, 25, or 26 for access to the virtual disk
- Configure hardware device registers that belong to the network hardware used by the Access program
- Make incorrect use of timer interrupts
- Open more than 127 files (that is, require the presence in the **config.sys** file of a **files** command with a value greater than 127)
- Use memory not assigned to them by the operating system.

The Access program can store DOS files on the host AIX system's fixed-disk drives. This ***virtual drive*** allows DOS to interface with the AIX system as if the AIX host were a fixed-disk drive.

You can use most DOS commands with files that the Access program stores on the AIX host. However, since AIX handles all disk-management functions for the virtual drive, the following DOS commands *cannot* be used for the virtual drive:

assign
chkdsk
diskcomp
diskcopy
fdisk
format
label
print
sys
tree.

Unpredictable results can occur if you issue one of these commands for the virtual drive or a file on the virtual drive.

Chapter 2. Using Host File Services

CONTENTS

About This Chapter 2-3
Beginning a File Services Session 2-4
  Logging in over a LAN Connection Path 2-5
  Logging in over an RS-232 Connection Path 2-6
  Logging in to Additional Hosts 2-7
  Logging in from the Command Line 2-8
Using the AIX File System 2-9
  AIX and DOS File Names 2-9
    Creating File Names with AIX Access for DOS Users 2-10
    Accessing Files with AIX Access for DOS Users 2-10
    Changing File Names 2-13
  Using AIX and DOS Text Files 2-13
    Converting AIX and DOS Files 2-14
  AIX File Protection 2-14
    Displaying File Modes 2-15
    Changing File Modes 2-15
    Effect of Directory Permissions on Files 2-16
    File Permissions with DOS Commands and Applications 2-16
    Record Locking and File Sharing 2-16
  AIX and DOS Commands 2-17
  Printing with AIX Access for DOS Users 2-17
Ending a File Services Session 2-20

About This Chapter

This chapter tells you how to use host file services, including:

- Beginning and ending a host-file services session
- Handling differences between DOS and AIX file naming conventions
- Converting text files between DOS and AIX formats
- Using AIX file permissions to protect your files in a multiuser environment
- Executing AIX processes from DOS
- Printing files on either a remote or local printer.

Note: The procedures described in this chapter, and the examples shown, assume that the Access program has been installed for your personal computer as described in Appendix A, "Installing AIX Access for DOS Users."

Beginning a File Services Session

A file services session allows you to run DOS applications, using and storing files on the host as if it were a local fixed disk. The following procedure starts a file services session.

1. If you are using an Access program working diskette, insert the diskette in drive A and close the drive door. If you are using a fixed disk, leave the drive A door open.

   Note: If you are using a diskette, make sure your working diskette #1 is set up for the type of network interface adapter installed in your personal computer. The type of network interface adapter should be indicated on the diskette label.

2. Switch on your personal computer. If the system unit is already on, press and hold the **Ctrl** and **Alt** keys, then press the **Del** key.

   A message similar to the following is displayed:

   ```
   AIX Access for DOS Users Version 1.1 (Serial #10.1779) Initialized
   ```

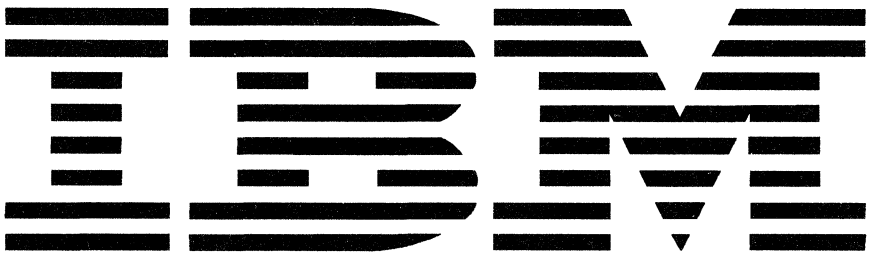
   Following this message, the system executes the commands from the **autoexec.bat** file.

   After the **autoexec.bat** commands complete execution, the system prompt is displayed. If you are using an Access program working diskette, the prompt is A>. If you are using a fixed disk, the prompt is usually C>.

3. At the prompt, enter the following:

   ```
   login
   ```

   Press **Enter**. The Access program logo screen is displayed as follows:



IBM AIX Access for DOS Users
Version 1.1

(C) International Business Machines Corporation 1988
(C) Locus Computing Corporation 1984, 1988

AIX is a Trademark of
International Business Machines Corporation

Press Enter (↵) to continue or Esc to Quit

4. Press **Enter** and the system briefly displays the following message:

   ```
   Searching for hosts
   ```

   Then an Access program host menu screen similar to the following is displayed:

```
                AIX Access for DOS Users
   Copyright (c) 1988 International Business Machines Corporation
        Copyright (c) 1984, 1988 Locus Computing Corporation
                    All rights reserved

Port                    Host Name
  1                     hera
  2                     apollo
  3                     athena
  4                     com1
  5                     com2



Please specify the number for the desired
connection port to your Host AIX system
and press enter:
```

The Access program host menu screen shows you the connected AIX hosts. When your personal computer is equipped with one or more RS-232 ports, each port is identified by com*n*, where *n* is a number from 1 to 4 depending on the number of ports installed in your computer.

When there are more hosts than can be displayed on the screen, the Access program displays only one screen of hosts. Additional hosts are not displayed. You can log in to other hosts by specifying the host name or RS-232 port on the DOS command line (refer to "Logging in from the Command Line" on page 2-8).

You can return to DOS by pressing and holding the **Ctrl** key and pressing the **C** key.

Logging in over a LAN Connection Path

Use the following procedure to begin an AIX host-file services session over a LAN.

1. At the host-selection prompt of the host menu, enter the port number that corresponds to your choice of hosts and press **Enter**. The Access program prompts you until you enter a port number listed on the screen.

   The following message briefly replaces the host-selection prompt:

   ```
   Connection Attempt in Progress....
   ```

   If the host you want is not available within about 10 seconds, the Access program displays the following message:

   ```
   Selected Host Not Available -- Try another? (y or n):
   ```

   Pressing `y` gives you the host-selection prompt, and pressing `n` returns you to DOS. You should not press **Enter** after the `y` or `n`.

   When a host connection is made, the Access program prompts you for your AIX user name as follows:

   ```
   Username:
   ```

2. Type your AIX user name and press **Enter**.

   The Access program then prompts you for your password as follows:

   ```
   Password:
   ```

3. Type your AIX user password and press **Enter**. Your password will not appear on the screen. This protects the security of your password.

   The Access program then attempts to establish a session with the AIX host you selected. If your password or user name was entered incorrectly, or if your user name is not valid on the selected host, the Access program cannot establish the session, and it displays the following message on your screen:

   ```
   Login incorrect
   Try Again? (y or n):
   ```

   Pressing `y` causes the Access program to prompt again for your user name and password. If you press `n`, the Access program displays `Login aborted` and returns you to DOS. You should not press **Enter** after the `y` or `n`.

When a file services session is established, a message similar to the following is displayed on your screen:

```
Welcome to AIX Access for DOS Users.
athena File System Available as Drive D:
```

The message indicates the AIX host associated with the selected LAN port (`athena` in this example).

You might see a different drive specified. The Access program assigns a drive specifier to the virtual drive that is higher than the highest current DOS drive specifier. For example, if the highest drive letter used by your personal computer is C, the drive specifier for the first AIX file system is D.

When the DOS system prompt is displayed, you can use the virtual drive as described in "Using the AIX File System" on page 2-9.

Logging in over an RS-232 Connection Path

Use the following procedure to begin an AIX host session over an RS-232 connection path.

1. At the host-selection prompt of the host menu, type the port number that corresponds to your choice of serial ports and press **Enter**. The Access program prompts you until you enter a port number listed on the screen.

   The following prompt replaces the host-selection prompt:

   ```
   Enter Baud Rate (300,1200,2400,4800,9600):
   ```

   Notes:

   a. The most efficient baud rate for transmitting data to the AIX host varies depending on your system configuration. In some arrangements, a 9600 baud can cause errors and result in performance degradation because of error recovery.

   b. If you are using modems to communicate with your host, choose the baud rate that matches the baud rate of your modems (see Appendix C, "Using a Dial-Up Modem" for more information).

   If you are uncertain about which baud rate to use, consult the person who administers your system.

2. Enter a baud rate from the list of choices, and press **Enter**.

   A five-line login window opens up at the bottom of the screen.

   Note: If you are using modems to communicate with your host, you can make the connection as described in Appendix C, "Using a Dial-Up Modem" using this window.

The AIX host banner and login prompt appear.

If the prompt does not appear, press **Enter** again. If the prompt still does not appear, then the baud rate of the host probably does not match the baud rate you selected. (You may see erroneous characters on the screen at this time.) If the host supports automatic baud rate on the RS-232 port, press the **F1** key repeatedly to change the host baud rate.

If the user name prompt does not appear, contact the person who administers your system for help.

You can press **F10** to return to DOS.

3. Type your AIX user name and press **Enter**.

   The AIX host then prompts you for your password as follows:

   ```
   Password:
   ```

4. Type your AIX user password and press **Enter**. Your password will not appear on the screen. This protects the security of your password.

   When you have successfully logged in, you will receive the AIX system prompt, normally a dollar sign ($).

5. Press the **F9** key to initiate the file services session. The following message is displayed:

   ```
   Connection Attempt in Progress....
   ```

   If a session cannot be established within about 10 seconds, the Access program displays the following prompt:

   ```
   Login incorrect
   Try Again? (y or n):
   ```

   Pressing `y` causes the AIX host to prompt again for your user name and password. If you press `n`, the Access program displays `Login aborted` and returns you to DOS. You should not press **Enter** after the `y` or `n`.

When a file services session is established, you will receive a message similar to the following:

```
Welcome to AIX Access for DOS Users
zeus File System Available as Drive D:
```

The AIX host attached through the selected RS-232 port, `zeus` in this example, is indicated in the message.

You might see different drive specifiers. The Access program assigns a drive specifier to the virtual drive that is higher than the highest current DOS specifier. For example, if the highest drive letter used by your personal computer is C, the drive specifier for the first AIX file system is D.

When the system prompt is displayed, you can use the virtual drive as described in "Using the AIX File System" on page 2-9.

Logging in to Additional Hosts

By default, the Access program is configured to support four virtual drives. Each virtual drive is assigned a different drive letter. For example, if the last local drive on your personal computer is the C drive, the four virtual drives are designated D, E, F, and G.

Note: In the example above, the drive letters D, E, F, and G are reserved for virtual drives. Do not use these drive letters for other drives when using the DOS **subst** command. Instead, choose a drive letter higher than that of the last reserved virtual drive letter.

The Access program can support a maximum of 16 virtual drives. For information about increasing the number of virtual drives your system supports, see "Changing the Maximum Number of Virtual Drives" on page 5-8.

To log in to additional hosts and establish additional virtual drives, repeat the login procedure described in the previous sections.

You may specify a virtual drive letter by using the **/d** option of the Access program **login** command. For instance, if your system is set up to support four virtual drives, and you have already established virtual drive D, you can specify drive E, F, or G as in the following example:

```
login /d:G
```

Note: If you do not specify a virtual drive letter when you log in, the system assigns the next available drive letter to the virtual drive for you.

When the maximum number of virtual drives for your system is reached, subsequent **login** commands display the following message:

```
login: no free virtual drives
```

See "Changing the Maximum Number of Virtual Drives" on page 5-8. for information about increasing the number of virtual drives your system supports.

Logging in from the Command Line

You can bypass the Access program logo screen and the host menu selection screen when you log in by specifying a host or RS-232 port with the Access program **login** command as in the following examples:

```
login apollo
```

or

```
login com1
```

You can either specify only the host name or RS-232 port and let the Access program prompt you for your user name and password, or you can include them on the command line as follows:

```
login host username password
```

where *host* is an available host name or RS-232 port.

If you log in through an RS-232 port, the Access program prompts you to enter a baud rate.

If you specify a host name that is not available through your LAN connection path, the Access program displays an error message as in the following example:

```
login: kronos not available.
```

If the host cannot be contacted within about 10 seconds, the Access program displays the following message:

```
login: connect failed
```

When a connection is made, the Access program attempts to establish a session on the host. If the Access program cannot establish a session, it displays the following message on your screen:

```
login: login failed.
```

When the session is established, a message similar to the following is displayed:

```
apollo File System Available as Drive D:
```

Remember the drive specifiers you will see differ depending on the highest drive letter currently used with DOS.

For information on ending a file services session, see "Ending a File Services Session" on page 2-20.

Using the AIX File System

Once the login process is complete, you can access the same AIX files as you do when you access the AIX host in any other way. These files are available to DOS on the virtual drive indicated in the connection message. When using the Access program you can do any of the following:

- Create or access files on the virtual drive using DOS commands or applications by specifying the virtual drive identifier (drive letter) as part of the file name.

  For example, to list a file called **memo** stored on the virtual drive D, you would enter the following:

  ```
  type D:memo
  ```

- Change your default drive to the virtual drive drive letter, as you would change to any real DOS drive.

  For example if the drive letter of the virtual drive is D, you would enter the following:

  ```
  d:
  ```

- Copy files, including DOS commands or application programs, between a local drive and the virtual drive using the same procedures as when you copy files between two local drives.
- Create directories on the virtual drive and set DOS search paths that specify directories on the virtual drive.
- Use the host file services of the Access program to access other users' files for which you have access permission. You can also make your files available to other users.
- Use DOS applications to manipulate files created by an AIX process or to create files for input to an AIX process.

When using the Access program, you should be aware of some basic differences between DOS and AIX. These differences concern the following topics:

- AIX and DOS file names
- AIX and DOS text files
- File protection
- AIX and DOS commands
- Printing with the Access program.

AIX and DOS File Names

The rules for naming files and directories differ between DOS and AIX. Before you begin working with files in the Access program you should become aware of these differences.

Case

AIX is case-sensitive, while DOS is not. Alphabetic characters in AIX file and directory names are usually lowercase, but any combination of uppercase and lowercase is also acceptable. Any combination you type is preserved.

DOS interprets all alphabetic characters in file and directory names as uppercase, whether you type them in uppercase or lowercase. While in AIX, **chapter1** and **CHAPTER1** are two different file names, DOS interprets them as the same file name.

Size

DOS limits file and directory names to eight characters plus an optional extension of up to three characters preceded by a period (.). The limit that AIX imposes on names varies depending on the AIX host. Ask the person who administers your system to find out the size limit on your host. In addition, an AIX name can contain periods anywhere in the name. For example, a file name such as **myfile.day** is acceptable to both AIX and DOS. However, file names such as **messagetoall**, **ver1.2.3**, and **chapter.seven** are acceptable only to AIX.

Characters

Both DOS and AIX accept alphabetic and non-alphabetic characters in file and directory names. However, AIX accepts more non-alphabetic characters than DOS. For example, control characters and spaces are valid characters in AIX names, though not in DOS names.

Directory Path Separators

In AIX, the individual directory names in a directory path are separated by a forward slash (/). In DOS, the directory names are separated by a backslash (\). When you use the the Access program **on** utility to execute AIX commands, you must specify any pathnames with AIX-style path separators (/).

Creating File Names with AIX Access for DOS Users

When you create files or directories during a file services session, file names must conform to standard DOS rules with respect to size and characters. You can type a file name with either uppercase or lowercase alphabetic characters. When you create the file on a local diskette or fixed disk, DOS converts all alphabetic characters to uppercase by default. However, when you create the file on the virtual drive, the Access program converts all alphabetic characters to lowercase. This prevents the creation of names such as **Chapter1** and **CHAPTER1**, which DOS cannot distinguish from **chapter1**. Any file you create with the Access program on the virtual drive can be accessed by either DOS or AIX.

Accessing Files with AIX Access for DOS Users

When you access files or directories during a file services session, you can type the file name with uppercase or lowercase alphabetic characters. DOS converts all alphabetic characters to uppercase when searching a local disk. The Access program converts all alphabetic characters to lowercase when searching the virtual drive.

When file names are displayed in DOS, they appear in uppercase. Entering the DOS command **dir** displays the contents of a directory on the virtual drive as in the following example:

```
Volume in drive D is zeus
Directory of D:\U\JONES

.              <DIR>       4-25-89       4:21p
..             <DIR>       4-18-89       4:04a
PCI            <DIR>       4-18-89       7:48p
REPORTS        <DIR>       4-16-89       3:04p
CHAPTER ONE       2478     7-22-89      10:08a
BDGTMEMO          3400     3-29-89       4:07p
STATUS  REP       2400     4-18-89       6:35p
PRJCTMEM          1465     3-31-89      11:14p
STAFF             3278     4-22-89       6:49p
           9 File (s)    151552 bytes free
```

The actual AIX names for these files are in lowercase. Names that appear to DOS to have file name extensions, like **chapter.one**, are displayed in the standard DOS manner with the extension listed in a separate column.

Note: When you add or delete small files (2 KB) on the virtual drive, you might not see a change in the number of bytes free displayed by **dir**. The number of bytes free for the virtual drive is calculated based on AIX blocks and can be off by plus or minus one block as a result of rounding up or down.

In general, you can access any file or directory in the AIX file system, whether it was created with the Access program or under AIX. However, you must use a special mapped name for AIX files or directories that do not conform to DOS rules. Files that do not conform include the following:

- Names longer than DOS allows
- Names with more than three characters following a period
- Names with more than one period
- Names that start with a period
- Names with non-alphabetic characters that DOS does not recognize
- Names with uppercase alphabetic characters.

When an AIX name that does not conform to DOS rules is accessed by a DOS utility or application, the Access program translates (maps) the name to a legal DOS name by appending a unique index consisting of an apostrophe followed by one, two, or three characters. If necessary, the AIX file name is truncated before the index is appended. For example, a file called **messagetoall** might be mapped to the name **mess'baq**. To determine the mapped name, issue the DOS command **dir**. In the following example, the displayed directory shows the mapped names for additional AIX directories and files called **Mail, messagetoall, message.tobob,** and **message_toted**. The DOS **dir** command shows only the mapped name, not the original AIX name.

```
 Volume in drive D is zeus
 Directory of D:\U\JONES

.              <DIR>          4-25-89      4:21p
..             <DIR>          4-18-89      4:04a
PCI            <DIR>          4-18-89      7:48p
REPORT         <DIR>          4-16-89      3:04p
CHAPTER   ONE       2478      7-22-89     10:08a
BGDTMEMO            3400      3-29-89      4:07p
STATUS    REP       2400      4-18-89      6:35p
PRJCTMEM            1465      3-31-89     11:14p
STAFF               3278      4-22-89      6:49p
MAIL'FPE       <DIR>          6-19-89      8:50p
MESS'BAQ            2400      6-18-89      6:35p
MESS'BBF  TOB        250      6-25-89      4:18p
MESS'BBS             598      6-30-89      6:01p
```

To display the contents of **messagetoall** using the DOS command **type**, enter the following:

```
type mess'baq
```

Although you use an AIX file's mapped name with DOS commands, you may also want to know the original AIX file or directory name. The Access program command **udir** displays the contents of an AIX directory in a form similar to that of the AIX **ls -l** command, showing both the AIX name and its corresponding mapped name.

For example, issuing the **udir** command for the directory **/u/jones** on the virtual drive results in the following display:

```
Volume in drive D is zeus
Directory of D:/U/JONES


.                       jones  drwxr-xr-x  <DIR>       4-25-89   4:21p
..                      root   drwxr-xr-x  <DIR>       4-18-89   4:04a
pci           PCI       jones  drwxrwxr-x  <DIR>       4-18-89   7:48p
report        REPORT    jones  drwxrwxrwx  <DIR>       4-16-89   3:04p
chapter.one   CHAPTER.ONE jones -rwxr-----        2478 7-22-89  10:08a
bdgtmemo      BDGTMEMO  jones  -rwxrw-r--        3400 3-29-89   4:07p
status.rep    STATUS.REP jones -rwxr--r--        2400 4-18-89   6:35p
prjctmem      PRJCTMEM  jones  -rwxr--r--        1465 3-31-89  11:14p
staff         STAFF     jones  -r--r-----        3278 4-22-89   6:49p
Mail          MAIL'FPE  jones  drwxr-----  <DIR>       6-19-89   8:50p
messagetoall  MESS'BAQ  jones  -rw-r--r--        2400 6-18-89   6:35p
message.tobob MESS'BBF.TOB jones -rw-r--r--         250 6-25-89   4:18p
message toted MESS'BBS  jones  -rw-r--r--         598 6-30-89   6:01p
      13 file (s)         148304 bytes free
```

As with the DOS **dir** command, you can specify specific file names or use the asterisk (*) or question mark (?) wildcard characters with **udir**. However, **udir** interprets these as DOS does, not as AIX does. For more on the **udir** command, refer to Chapter 7, "Command Summary."

Changing File Names

When it is inconvenient to work with the mapped name, you can use the Access program **on** command to run the AIX **mv** command and rename files when the source or destination file names are not valid DOS names. For example, if you have created several files such as **message1**, **message2**, and **message3** on the virtual drive D, and would like these files to have more descriptive AIX names, you can rename them by entering:

```
on d: mv message1 messagetobob
on d: mv message2 messagetojohn
on d: mv message3 messagetosteve
```

The AIX **mv** command is similar to the DOS **ren** command, but provides you with the additional ability to rename directories. For example, to avoid having to use the mapped name, you can rename an AIX directory called **directmail** with a valid DOS name by entering:

```
on d: mv directmail dmail
```

As illustrated, use the full AIX name, not the mapped name, when using the **mv** command.

Finally, you can also use the AIX **mv** command to move a file from one directory to another on the virtual drive, as the following example shows:

```
on d: mv /u/jones/report/budget /u/jones/memos/budgetmemo
```

Note: You must use AIX-style path separators (/) with the **on** command.

To avoid prefixing the **mv** command with **on**, you can link **on.exe** and **mv.exe** as described in "Linking the on Command to .exe Files" on page 3-6.

Using AIX and DOS Text Files

DOS and AIX differ in the way each stores text files. AIX stores text lines as a sequence of characters terminated by a newline character. DOS terminates text lines with a carriage return character, followed by a line feed character. A file created in one format appears corrupted when accessed by the other.

For example, suppose you have the following file in AIX format:

```
Dear Bob:

This letter is in response to your inquiry of 25 February.
The information you requested is enclosed, along with other
materials that should help you evaluate our product line.
```

If you listed this file with the DOS **type** command, you would see the following:

```
Dear Bob:

      This letter is in response to your inquiry of 25 February.
                                                                T
he information you requested is enclosed, along with other
                                                        materials
that should help you to evaluate our product line.
```

Similarly, if you had created the original file in DOS format and listed it with an AIX **cat -v** command, you would see the following:

```
Dear Bob:^M
^M
This letter is in response to your inquiry of 25 February.^M
The information you requested is enclosed, along with other ^M
materials that should help you evaluate our product line.^M
^Z
```

Converting AIX and DOS Files

You can use any file created during an Access program file services session just as you would any DOS file, since the files created on the virtual drive are stored in DOS format.

However, to use a text file in AIX format with DOS programs in your file services session, you must convert the file to DOS format using the Access program **aix2dos** command. The following converts the AIX file **letter** to a DOS file **ltr.dos:**

```
aix2dos  letter  ltr.dos
```

You can also convert the file and copy it from the virtual drive to a local drive on your personal computer in one step, as in the following:

```
aix2dos  D:bdgtmemo  a:budget
```

During the file services session, when you create DOS text files that you want to use later in an AIX session, you can convert them to AIX format with the Access program **dos2aix** command. For example, the following example creates an AIX file **memo.aix**.

```
dos2aix  memo.dos  memo.aix
```

When you use the **aix2dos** or **dos2aix** commands, you use DOS file names. You cannot specify the same name for the source file and the target file. Also, you are not allowed to redirect your output back into the source file.

If you omit the second file name, the converted text is displayed on your screen, but it is not saved in a file. Use the **aix2dos** or **dos2aix** commands to convert a file to the format you need. If the file is already in the format you need, using the conversion command is not harmful.

For more on **dos2aix** and **aix2dos**, refer to Chapter 7, "Command Summary."

AIX File Protection

Unlike most DOS systems, the AIX system can be accessed by more than one user. Therefore, it provides tools to prevent inspection, alteration, or execution of files by unauthorized users. The owner (usually the creator) of each AIX file can grant or deny himself, other users in his user group, or all AIX users permission to read, modify, or execute his files.

All the files and directories you create or access on the virtual drive during an Access program file services session are protected by these permission assignments. The term ***permission mode*** (or ***mode***) refers to the specific protection for a file or directory, as described in the next section.

Displaying File Modes

When you create a file or a directory on the virtual drive, it inherits the default permission mode set by the **UMASK** environment variable from the AIX root. You can check the current mode of any file or directory by issuing the **udir** command to display the file's permission mode. The third column of the display shows the owner, and the fourth column shows the current mode as a 10-character string in the following form:

```
drwxrwxrwx
```

The first character indicates whether the entry is one of the following:

- file (-)
- directory (d)
- block device (b)
- character device (c)
- symbolic link (s)
- named pipe (p).

The remaining nine characters indicate the read, write, and execute permissions in groups of three. That is, the first set of three characters represents the permissions for the file or directory owner, the second set of three the permissions for others in the owner's user group, and the last set the permissions for everyone else.

r	Specifies permission to display or read the file.
w	Specifies permission to modify or write the file.
x	Specifies permission to execute the file as an AIX program.
-	Indicates that the particular permission is denied.

For more information on setting the **UMASK** environment variable, contact the person who administers your system.

Changing File Modes

You can change the permission modes for your files and directories on a virtual drive by using the Access program **on** command to run the AIX **chmod** command.

Note: You must use the Access program **on** command rather than the DOS **attrib** command to change the permission modes of files on a virtual drive. The **on** command is described in detail in Chapter 3, "The on Utilities."

For example, suppose you want to protect a file on virtual drive D called **budget** so that no one, not even you, can modify it. You would issue the following **on** command:

```
on d: chmod a=r budget
```

Where `d` is the virtual drive on which the AIX **chmod** command is to be run, `a` is the group your change affects, and `r` is the permission you are changing. You can specify any of the following groups:

- **u** for the owner (user)
- **g** for members of the owner's user group
- **o** for others on the system
- **a** for all three groups.

The previous example sets all three groups to read-only permission.

Alternately, you can indicate only the permissions you want to change without having to specify all the permission characters. To change the write permission so you can modify your **budget** file, you could enter the following:

```
on d: chmod u+w budget
```

To give write and execute permission to others in your user group as well as everyone else on the system, you could enter the following:

```
on d: chmod go+wx budget
```

In this form of the AIX **chmod** command, you first specify the groups your change affects, followed by a plus (+) or minus (−) sign (depending on whether you are granting or denying the permission), followed by the permissions you are changing.

You can link **on.exe** and **chmod.exe** as described in "Linking the on Command to .exe Files" on page 3-6, allowing you to type the **chmod** command without prefixing it with **on**.

Effect of Directory Permissions on Files

You must have execute permission for a directory on the virtual drive in order to access any of the files in that directory. Without execute permission you *cannot* do any of the following:

- Copy files from or into the directory
- Make the directory your working directory
- Rename, inspect, or modify any file in the directory.

To provide access to files, be sure you assign the appropriate permissions to your directories. If you want to completely deny access to files in a directory without changing the mode for each file, remove execute permissions from the directory.

File Permissions with DOS Commands and Applications

It is important to note that DOS does not have the same system of file permissions as AIX. This can create unexpected results when DOS commands interact with files on the virtual drive.

When you execute DOS programs stored on the virtual drive, DOS does not reference the execute permission for the file. Instead, DOS identifies executable files by their file name extensions, such as **.com**, **.exe**, or **.bat**. You must have read permission for DOS executable files, and execute permission for the directories they are in.

Sometimes the message DOS returns depends on the file permissions. For example, when a DOS command you issue encounters a file for which you do not have read access, DOS displays the message `File not found`, even though the file exists. Similarly, if you try to create a file in a directory for which you do not have write access, DOS displays the message `File creation error` or `Directory full`.

Certain DOS applications, when used with the Access program in a multiuser environment, do not protect files from being simultaneously updated by any users with write permission. This is because these applications do not support file locking and record locking. These types of applications include those that update files by first renaming the original version (usually to a name with a **.bak** extension) and then creating a new version with the original name.

When working with applications that do not support locking, you should carefully consider which combination of file and directory permissions give the most appropriate protection. For example, to prevent a file you are working on from being updated by another user, you could temporarily remove all other users' read and write permissions for that file. This prevents anyone from altering the file until you have finished your current task.

Record Locking and File Sharing

If you are using applications that support record locking and file sharing, file permissions are less important. The Access program supports standard IBM DOS file-sharing and record-locking conventions. ***File sharing*** means that an application can open a file so that other applications can also access the file. Similarly, if you open a file with an application that supports record locking, such as a database management program, no one else can access a record while you are working on it.

Not all applications support file sharing and record locking. The following discussion applies only to those that do.

When installing network versions of certain DOS applications, you are prompted to choose a network type. It is recommended that you try **IBM PC Network** or another compatible network type first. You might have to experiment to find the selection that best supports file sharing and record locking.

You should not use the **copy** or **backup** command with files on the virtual drive that another user is accessing. If there are locks on the files (or records in the files), DOS

returns an error message indicating that the **copy** or **backup** command failed. In fact, an incomplete copy was made.

If a file appears to be locked when there are no other users logged in, or if access is allowed to a file that should be locked, there could be a problem with file sharing and record locking, and you should notify the person who administers your system.

Notes:

1. Record locking and file sharing only apply to DOS programs. Any AIX user with the appropriate AIX file permissions can read and write on the virtual disk at any time, regardless of its DOS lock status. For instance, a person using the Access program can read the records of a DOS lock file at any time by using either terminal emulation or the **on** command.
2. The IBM System/370 host computer does not support DOS record locking.

AIX and DOS Commands

Users familiar with both DOS and AIX may find occasions during a host-file services session when they would like to use an AIX command or process instead of a DOS command or application. The Access program gives you an environment that allows you to use both DOS and AIX.

The Access program **on** utility lets you issue a non-interactive AIX command from within your host-file services session without having to switch to terminal emulation. For example, to generate a list of users logged in to the AIX system and display the output on your personal computer, you can execute the AIX **who** command as follows:

```
on - who
```

For more information on using the **on** command to access AIX commands, refer to Chapter 7, "Command Summary."

Printing with AIX Access for DOS Users

During a file session you can print files from both the virtual drive and local drives. You can also direct printing to a local printer attached to your personal computer or a remote AIX printer.

When you first enter the Access program host-file services session, by default, all your printing is directed to your local printer. You can print files stored on local drives or virtual drives using any of these standard DOS printing methods:

- Pressing the **Print Screen** key (**Shift-PrtSc** for PCs, ATs, and XTs) to print the current screen contents.
- Pressing the **Ctrl** and **Print Screen** keys (**Ctrl-PrtSc** for PCs, ATs, and XTs) together to print continuous screen contents.
- Issuing the DOS **print** command and specifying the appropriate file name.
- Using the DOS **copy** command to copy a file to the local printer device (PRN).
- Invoking the print function from within a DOS application.

In addition to printing on your local printer, you can send your print requests to a remote printer connected to the AIX system. To direct printing to any of the three remote print streams (LPT1, LPT2, or LPT3), specify the print stream and an AIX host name with the Access program **printer** command.

There are three ways of specifying the AIX host for remote printing:

- Refer to the host by the host name. For example, to direct LPT1 print requests to the system printer of an AIX host called `frodo`, enter:

```
printer frodo lpt1
```

- Refer to the host by the virtual drive letter associated with it. For example, you can use the virtual drive letter to identify the host as follows:

```
printer d: lpt1
```

- Use a hyphen (-) to signify the host associated with the current working drive (the host on which you are working when you issue the **printer** command). For example, to direct LPT1 print requests to the system printer of the host associated with the current working drive, enter the following:

```
printer - lpt1
```

In all cases when you omit the print stream identifier, LPT1 is assumed.

When you specify remote printing with the the Access program **printer** command, you can use the following print methods:

- Using **Print Screen** (**Shift-PrtSc** for PCs, XTs, and ATs) to print the current screen contents. When you use this method, printing is always directed to the LPT1 print stream.
- Pressing **Ctrl-Print Screen** (**Ctrl-PrtSc** for PCs, ATs, and XTs) to print continuous screen contents. If you use this print method, you must first use the **printer /x0** command. For more information on the **printer /x0** command, see "printer" on page 7-25.
- Using the DOS **copy** command to copy a file to one of the three print streams LPT1, LPT2, or LPT3.
- Invoking the print function from within a DOS application.

Note: Do not use the DOS **print** command to initiate printing after you issue the **printer** command.

Where and how each remote print job prints depends on the configuration of the AIX **print** command on the remote host. Under AIX, each **print** command causes output to be sent to a particular printer. In addition, AIX allows you to define printer programs that send output to specific printers. For example, the program **enscript** may send output to a laser printer, while **lp** might direct output to a line printer.

The Access program **printer** command allows you to specify an AIX command or program for processing each of the three DOS print streams. For example, if you entered the following:

```
printer e: lpt3 enscript
```

all subsequent print requests directed to LPT3 are processed by the **enscript** program on the host associated with virtual drive E and printed on the printer specified by the **enscript** program. Use the **/r** option with the **printer** command to reset the print stream to go to the default print handler. For example:

```
printer e: lpt3 /r
```

The AIX program specified with the **printer** command does not have to send output to a printer. When you specify commands or programs with the **printer** command, you can include any valid options, as well as pipe and redirection symbols. If the command string contains spaces, it must be enclosed in quotes as in this example:

```
printer - lpt2 "spell | /usr/pci/bin/aix2dos > sp.err"
```

This causes output directed to LPT2 to be sent to the AIX spelling check program, with the errors converted to DOS format and written to a file called **sp.err**.

If you do not specify a command or program in the **printer** command, the Access program sends print requests for that print stream to the AIX **print** command for output to the

default system printer. Alternately, you can establish your own defaults as described in Chapter 5, "Tailoring AIX Access for DOS Users."

When you invoke remote printing, the Access program buffers your print requests but does not send them to the print command until you close the print stream. You can close the print stream by exiting the DOS application program or issuing another DOS command. To speed the processing of print requests, you can specify a time-out period using the **/t** option with the **printer** command. For example, to request print processing after 10 seconds for requests sent to LPT2 on the current default drive, enter the following:

```
printer - lpt2 /t10
```

You can specify time-out periods from 1 to 3,600 seconds. If you specify the **/t** option without specifying a number of seconds, the default time-out of 45 seconds is used.

Note: Do not make the time-out value too small. If the value is too small, printing can be truncated or interspersed with other print jobs.

If you specify **/t0**, printing is not initiated by time-out. Specifying **/t0** is equivalent to omitting the **/t** option.

To prevent buffered print requests from processing until a specific time use the **/x0** option. For example, to hold all print requests directed to print stream LPT3 on the current default drive, enter the following:

```
printer - lpt3 /x0
```

If you use **Ctrl-Print Screen** (or **Ctrl-PrtSc** for PCs, XTs, and ATs) to print continuous screen contents, you should first issue a **printer** command with the **/x0** option to turn off automatic processing of your print requests on program exit. Otherwise, the Access program closes and prints your accumulated screen output after each command exits, rather than printing your entire session in one print job. When you are finished recording screen contents for all commands, press **Ctrl-Print Screen** again (or **Ctrl-PrtSc** for PCs, XTs, and ATs) and issue the **printer** command with the **/p** or /x1 option to cause your accumulated output to be printed.

When you are ready to process these print requests, enter the following:

```
printer lpt3 /p
```

If you decide not to process buffered print requests, you can delete them by entering the following:

```
printer lpt3 /d
```

To switch back to automatic processing of print requests (the default when you enter an Access program host-file services session), enter the following:

```
printer lpt3 /x1
```

You can switch back to local printing at any time during your host-file services session by entering:

```
printer local lpt3
```

If you are not sure whether your printing is directed to your local printer or to a remote printer, enter the following:

```
printer
```

The Access program displays the current printer setting and options in effect for each of the three print streams.

For more information on the **printer** command, refer to Chapter 7, "Command Summary."

Ending a File Services Session

Use the following procedure to end host-file services sessions.

1. Change the current drive back to your original drive. For example, enter:

   ```
   a:
   ```

   If you are working from a diskette, then you probably started from drive A, and if not, then you probably started from drive C.

   You may also change the current drive to a virtual drive associated with a host session that you are not going to end.

2. Enter the the Access program **logout** command in one of the following forms:

   ```
   logout
   ```

   to end all file services sessions.

   ```
   logout hostname
   ```

   to end the session associated with the specified host.

   ```
   logout driveletter:
   ```

   to end the session on the host associated with the virtual drive specified by *driveletter*.

Note: If your current drive is a virtual drive that has been disconnected, the following DOS system prompt is displayed:

```
Current drive is no longer valid>
```

If this happens, change the current drive to some valid drive, such as the original drive (either A or C).

The Access program disconnects you from the AIX hosts and displays a message showing the virtual drive letters, and associated host names from which you disconnected.

You can continue to use DOS and Access program files and programs located on your personal computer's diskettes, your personal computer's fixed disks, or virtual drives associated with AIX hosts from which you have not disconnected. However, any search paths previously set (for example, in **autoexec.bat** at initialization) remain in effect. If a search path refers to a disconnected virtual drive, you should delete the virtual drive letter from the search path.

Chapter 3. The on Utilities

CONTENTS

About This Chapter 3-3
Uses of the on Utilities 3-4
Using the on Command 3-4
  The on Command and .exe Files 3-5
  Linking the on Command to .exe Files 3-6
  Setting Environment Variables 3-7
  Expanding DOS Environment Space 3-8
  Breaking Out of on 3-8
Job Control 3-9
  The Job Table 3-9
  Detaching Tasks 3-9
  Keeping Track of Detached Tasks 3-10
  Reattaching to a Detached Process 3-11
  Saving Output from Completed Jobs 3-12
  Stopping Detached Jobs 3-12
Pipes and Redirection 3-13
  DOS Pipes 3-13
  DOS Output Redirection 3-13
  DOS Input Redirection 3-13
  Using Pipes and Redirection in Detached Tasks 3-14
Summary of Restrictions and Cautions 3-15

About This Chapter

This chapter discusses the three DOS programs that comprise the **on** utilities:

- the **on** program
- the **jobs** program
- the **kill** program.

In addition, this chapter discusses how to set environment variables, control jobs, and work with pipes and redirection. A summary of **on** restrictions and cautions is also provided.

Uses of the on Utilities

The **on** utilities allow personal computer users to execute AIX commands and programs on specified AIX host computers and view the output on their personal computer. You can extend the function of DOS by executing AIX commands on DOS files as if they were DOS commands.

With the **on** utilities you can execute AIX commands when terminal emulation is not available. You can also execute a short sequence of AIX commands without logging in to a host, even when terminal emulation is available. This uses fewer resources on the host computer and increases overall system performance.

The **on** utilities can only be used to execute noninteractive AIX commands and programs. Noninteractive commands are those that do not initiate a conversation with the user. For example, **on** cannot run the interactive AIX text editors **vi** and **ed**. A terminal emulation session is required for interactive programs. However, **on** can be used with AIX stream editors like **sed**. The Access program also includes a DOS version of the **vi** interactive editor, which is described in Chapter 6, "The AIX Access for DOS Users vi Editor."

Using the on Command

To execute an AIX command using the **on** utility, type `on` followed by a specified host and AIX command. You may specify hosts in the following three ways:

1. By using a LAN-connected host name or RS-232 port name
2. By using a virtual drive letter
3. By using a dash (-) to specify the current host.

You must have already established a file services session with the specified host.

You can specify an AIX host by name if it is connected through a LAN, as in the following example:

```
on oberon cal 1989
```

In this example, `oberon` is the name of a host connected over a LAN. The host name must be entered exactly as shown on the host-selection menu. The AIX **cal** command will display a calendar of 1989 on your personal computer terminal.

If you are connected to a host over an RS-232 connection path, you would specify the host as follows:

```
on com1 cal 1989
```

In this example, `com1` is the RS-232 port connected to the host.

You can specify an AIX host by using the virtual drive letter associated with the host as in this example:

```
on D: cal 1989
```

Here, `D` is the virtual drive letter assigned to the host when you established the file services session.

If your current drive is a virtual drive, and you want to execute the **cal** command on that host, enter the following:

```
on - cal 1989
```

If you attempt to use this form of **on** when your current drive is not a virtual drive, you will receive an error message.

AIX commands executed using **on** can contain any flags or parameters that are valid for the command. Specify AIX file names with their AIX names, not with their mapped DOS names. You must also use AIX-style directory separators (/) with the AIX command.

The Access program **aix2dos** utility is built in to **on**. When **on** executes an AIX command, it automatically converts the output from the command from AIX format to DOS format.

If **on** cannot execute the requested AIX command, either because it cannot find a requested file, or because you do not have execute permission for the requested file, **on** returns the following error message:

```
command: access denied or file not found
```

where *command* is the name of the AIX command you attempted to run.

Notes:

1. The **on** utility cannot execute multiple AIX commands separated by semicolons. Use only individual AIX commands with **on**.
2. By default, the **on** command executes the AIX shell on the host. For information on running other shells, see "The ONPREFIX Environment Variable" on page 5-10.

The on Command and .exe Files

You can execute AIX commands from DOS without prefixing the command name with on by making a copy of the DOS file **on.exe** that has the name of the AIX command with a **.exe** extension. For example, to make a **.exe** file for the AIX **ls** command, change directories to where you installed the file **on.exe** and enter the following:

```
copy on.exe ls.exe
```

Set your current drive to a virtual drive on a host where you have already established a file services session. Then you can execute **ls** on the host as follows:

```
ls -l
```

AIX commands executed like this can be followed by any flags or parameters that are valid for the original AIX command. Specify AIX file names with their AIX names, not with their mapped DOS names. You must also use AIX-style directory separators (/) with the AIX command.

Your current drive must be set to a virtual drive on a host where you have already established a file services session. To run the command on a specific host, you must set the current drive to the virtual drive associated with the host. For example, entering **ls -l** to invoke the program **ls.exe** is the same as entering:

```
on - ls -l
```

This feature allows you to use AIX utilities on DOS files stored on virtual drives as if they were DOS utilities. For example, to search DOS files for particular character strings and display the ends of the DOS files, you would first create **.exe** files for the AIX **grep** and **tail** commands by moving to the directory where **on.exe** is located and entering the following:

```
copy on.exe grep.exe
copy on.exe tail.exe
```

Then you could invoke them as follows:

```
grep -n   Wilson letter.txt
tail -23  letter.txt
```

The first command reports every line in **letter.txt** that contains the name Wilson, and the second command display the last 23 lines of **letter.txt** on your personal computer screen.

When making **.exe** files with **on**, the following restrictions apply:

- The AIX command must contain only lowercase characters in its original AIX form.
- The name of the AIX command cannot violate any of the DOS naming conventions. For example, you cannot create a **.exe** version of a shell script named **mycalendar** because its name contains more than eight characters.
- You cannot make files with the names **jobs.exe** and **kill.exe**. These names are reserved for the Access program. If you want to execute the AIX commands with these names, use the prefix form of **on** as in the following example:

```
D> on - kill 247
D> on - jobs
```

  Note: It is recommended that you use the Access program **kill** command to stop background processes, rather than the AIX **kill** command.

- You cannot make **.exe** files with the names of DOS internal commands. Some DOS and AIX commands have the same names, but perform very different functions. To run an AIX process with the same name as a DOS internal command, use the prefix form of **on** as in the following example:

```
D> on - type letter.txt
```

Linking the on Command to .exe Files

You can use the DOS **copy** command to create a library of **.exe** files for executing AIX processes, as described in "The on Command and .exe Files" on page 3-5, and place them all in a directory specified in your DOS search path. However, this method uses disk space.

A better idea is to use the AIX **ln** command to create a library of linked **.exe** files. The advantage of using AIX **ln** over DOS **copy** is that **ln** allows a file to have more than one name without taking up extra disk space.

In order to use **ln**, a copy of **on.exe** must be stored on an AIX host on a virtual drive. The linked **.exe** files must also be stored on the virtual drive.

If you create a public directory on an AIX host for the **.exe** files, then Access program users can establish file services sessions, set their DOS search paths to the public directory, and use the AIX processes to extend the functionality of their DOS sessions.

Use the following procedure to create a library of AIX functions:

1. Use the **login** command to establish a file services session and create a new directory (the library) on the virtual drive.
2. Copy **on.exe** from where you installed it on your local drive to the new directory on the virtual drive. Change drives to the virtual drive and change directories to the new directory.
3. Create AIX library functions in the new directory by linking **.exe** files to the **on.exe** file.

   For example, to make the library function **ln**, link a file named **ln.exe** to the **on.exe** file using the **on** command as follows:

```
on - ln on.exe ln.exe
```

   Doing this link first allows you to use **ln.exe** to call **on** for making additional links. For instance, you can create **grep** and **tail** library functions as follows:

```
ln on.exe grep.exe
ln on.exe tail.exe
```

   The library now has the following three functions: **ln**, **grep**, and **tail**. They can be used like the AIX versions, just as **ln** is used here.

Library functions can be used on AIX files as well as on DOS files on your local drives or any virtual drives. Any flags or parameters valid for the equivalent AIX programs can be used with these functions.

Notes:

1. You must have your current drive set to the virtual drive where the library is located whenever you want to use the library functions. However, your current directory does *not* have to be set to the library directory if you set your DOS search path to include the library directory.
2. You must use AIX-style pathnames with these functions (AIX directory separators (/) and file names, and not DOSmapped names).
3. You cannot use **ln** to link copies across AIX file system boundaries. See **cp** in *AIX Operating System Commands Reference* for more information on **ln**.

Setting Environment Variables

When you use **on** to execute an AIX command, the AIX system searches for the command according to the AIX **PATH** environment variable. The current directory used by the AIX process that gets executed is the current directory of the virtual drive you selected. For instance, if the virtual drive is D, and the current directory of D is **/u/bin**, then the current directory for the AIX process is **/u/bin**.

The **on** program uses the following two DOS environment variables to alter the AIX environment:

UPATH Specifies an alternate AIX search path. Its format is the same as the AIX shell **PATH** variable, with path names separated by colons. For example, you might change the default AIX search path by using the DOS **set** command as follows:

```
set upath=:/u/bin:/u/ucb:/bin:/etc:/usr/bin
```

Use slashes (not backslashes) as path separators in the **UPATH** environment variable.

EXPORT Contains a list of DOS environment variables that are to be duplicated in the AIX environment. For example, you can put the following variables to your DOS environment:

```
set HOME=/u/paula
set TZ=PST8PDT
set LPDEST=laser2
```

You can set these variables in the AIX environment by setting the DOS **EXPORT** environment variable as follows:

```
set export=home tz lpdest
```

Because DOS environment variable names are uppercase, only uppercase variable names can be exported to the AIX environment. The **on** utility converts variable names to uppercase as they are exported. This means you can type everything to the left of the equal sign in the DOS **set** command in either uppercase or lowercase. For example, the following commands are equivalent:

```
set lpdest=laser2
set LPDEST=laser2
```

By contrast, **on** passes everything to the right of the equals sign literally. Therefore, you must enter the values of DOS variables that will be exported to the AIX environment exactly as you would enter them at an AIX prompt. For example, if an AIX host has a printer named `laser2` (lowercase only) then the following sets the **LPDEST** variable correctly:

```
set lpdest=laser2
```

But the following does *not*:

```
set LPDEST=LASER2
```

since `LASER2` remains uppercase.

To view variables that have been exported to the AIX environment, enter the following:

```
on - set
```

Expanding DOS Environment Space

By default, DOS sets aside only 160 bytes of memory to contain all of the characters in all DOS environment variables. The Access program user can run out of DOS environment space, for example, simply by setting any of the following:

- A DOS search path containing long virtual-drive path names
- A **UPATH** variable with several long path names
- A long list of variables to export to the AIX environment.

Place the following line in the DOS **config.sys** file to reserve additional environment space:

```
shell=driveletter: command.com /e:nn /p
```

where *driveletter* is the drive ID letter of the fixed disk or diskette from which you bring up your system, and where *nn* is an integer ranging from 160 to 32,768 representing the number of bytes of memory to reserve.

Breaking Out of on

You can interrupt the **on** command using **Ctrl-C** or **Ctrl-Break**, unless it is running as a ***detached process***. (A detached process is a process placed in the background of the AIX environment. Refer to "Job Control.") After you use **Ctrl-C** or **Ctrl-Break**, the system will prompt you as follows:

```
a - abort, c - continue, d - detach:
```

If you do not select one of the three options, the system will continue to prompt you.

Pressing `a` kills the job, clears the job from the job table, and returns you to a DOS prompt. Pressing `c` allows the process to continue as before, in the DOS foreground. Pressing `d` detaches the AIX process from your terminal, returns your DOS prompt, and continues to run the specified program in the background under AIX. For more information on detached processes and job tables, see "Job Control."

Job Control

The **on** utilities include an AIX process-tracking feature that allows you to do the following tasks on all AIX processes initiated by the **on** command:

- Start one or more AIX processes from your DOS prompt and run them in the background.
- Detach an AIX process that is currently running in the foreground so that it continues to run in the background.
- View a job table showing the status of all currently executing and completed AIX processes initiated by the **on** command.
- View the output of a detached process, including commands that have completed execution.

The following sections describe the features for controlling AIX jobs executed by **on**.

The Job Table

The Access program maintains a job table that keeps track of detached AIX processes initiated by **on**. The job table shows the status of the detached **on** commands.

The job table holds one entry for each **on** command. However, each AIX command can create several subordinate processes. For example, one **spell** command takes only one space in the DOS job table, but is displayed in an AIX **ps** (process status) list as four or five processes.

All AIX systems limit the number of simultaneous processes each user can run. If several AIX commands initiated with **on** each create several processes, the limit for AIX processes per user can be reached before the limit for entries in the job table.

Detaching Tasks

You can detach AIX processes initiated with **on** so they run in the AIX background by using either of the following two methods:

1. Initiate an **on** command and interrupt it with **Ctrl-C** or **Ctrl-Break**. When prompted, change the process to become detached.

   For example, if you issue an **on** command and interrupt it with **Ctrl-C** before the DOS prompt returns, the following prompt is displayed:

   ```
   a - abort, c - continue, d - detach
   ```

   When you type `d`, the process is placed in the background under AIX.

2. Add an ampersand (&) to the end of the **on** command. For example:

   ```
   D> on - spell memo &
   ```

   The ampersand must be preceded by a space.

In both cases, **on** responds with a message in the following form:

[jobnumber] pid

The *jobnumber* is the number of the job in the DOS job table, and *pid* is the AIX process ID number returned by AIX. Your DOS prompt then returns and you can issue additional commands (including **on** commands) while the detached process continues executing in the background.

Any output produced by a detached **on** process (for example, the spelling errors found by **spell**) is stored by default in a temporary file. You can see the contents of this temporary

file by reattaching to the process. For information regarding reattaching to a process, see "Reattaching to a Detached Process" on page 3-11.

You can also use pipes and redirection to send the output of detached processes to any program or file you choose. (See "Pipes and Redirection" on page 3-13.)

It is important when initiating detached processes to remember the following points:

- If the job table is full at the time you attempt to start an AIX process, **on** returns an error message in the following form:

  cmd: Job table full

  The AIX command is specified by *cmd*.

  The error message is returned when every position in the job table is filled, whether the entries are completed (Done) or still executing (Running).

- If the maximum allowed number of AIX processes was reached when you issued an **on** command, **on** returns this error message:

  AIX exec failed

  This message can occur when there are still places to fill in the job table. The total number of AIX processes you can start is set by AIX independently of the maximum number of job-table entries available.

Keeping Track of Detached Tasks

You can use the **jobs** command to perform any of the following processes:

- Display job-table information
- Clear the job table of entries for completed jobs
- Reattach to detached jobs.

Entering **jobs** displays the current job table in the following format:

```
JOB    HOST        STATE       EXIT STATUS   COMMAND
[1]    d:host1     Running                   AIXcommand1
[2]    e:host2     Done        exit(0)       AIXcommand2
```

The five columns in the job table report the following information:

JOB Shows the job ID number that **on** assigned to each process when the process was detached.

HOST Identifies the virtual drive and host name.

STATE Indicates whether the process is running or done.

COMMAND Shows the AIX command that was requested.

The **EXIT STATUS** column is blank while the job is running. When the job has finished executing, the **EXIT STATUS** column contains one of the following:

exit(*nn*) The job terminated with an exit status of *nn*. An exit status of 0 means the process terminated normally. Any other value can indicate an abnormal termination of the process.

unknown The job terminated, but **on** was unable to determine its exit status.

signal(*nn*) A signal was received that killed the process. In this case, *nn* indicates the signal received.

coredump(*nn*) The signal received caused a core dump to occur. This is a special case of **signal**. In this case, *nn* indicates the signal received.

err3(*nn*) An error in the functioning of **on** or the Access program has occurred. In this case, *nn* indicates the AIX error number.

Completed processes remain in the job table until you clear them by entering **jobs** with a single hyphen as a parameter. To clear the job table, enter the following:

```
D> jobs -
```

When you enter the command to clear the job table the following events occur:

- The current status of the job table is displayed, including all currently Done entries.
- The Done entries are cleared from the job table.
- Any temporary files in the /**tmp** directory associated with Done jobs are removed.

Once jobs are cleared from the job table, they cannot be reattached. Output from any process being saved in temporary files for reattachment and review is discarded.

If the job table is empty when you enter **jobs**, the following message is displayed:

```
jobs: no remote processes in Job table
```

The job table is empty when no detached jobs have yet run or when you have cleared the job table of all entries.

Reattaching to a Detached Process

Using the **on** utilities, you can reattach a process that has been detached. Reattaching allows you to:

- View all previous output of a currently running program and continue to view new output as the program sends it to your screen
- View the output of a program that has finished running.

To reattach either to a currently running job or completed job, use the **jobs** command in the form:

```
jobs %jobnumber
```

The *jobnumber* is the job number of the detached process. For example, to start the AIX **spell** program as a detached process, enter the following:

```
on - spell memo &
```

When you enter this command, **on** displays a job number and AIX process ID similar to the following:

```
[1] 4376
```

To reattach to your **spell** job, enter the following command:

```
jobs %1
```

If you decide to reattach to a job and do not know the job number, enter the **jobs** command without any parameters. You will receive a display similar to the following:

```
JOB    HOST        STATE       EXIT STATUS   COMMAND
[1]    d:host1     Running                   spell memo
[2]    e:host2     Done        exit(0)       who
```

Specifying the percent sign without a job number reattaches you to the lowest numbered process in the job table:

```
jobs %
```

When you reattach to a currently running job (whose state is Running in the job table), you see all the output produced by the job up to the time you reattach to it. The job then continues to run, and you see any additional output as it is printed. You can detach the process again at any time while it is running. Each time you reattach it, you see *all* output printed up to the time you reattach.

When you reattach to a completed job (whose state is Done in the job table), you see all output printed by the job. Once you reattach to a completed job, the temporary file storing the output is removed, and the job-table entry for that job is cleared. If the job completes execution while it is attached, the temporary file is also removed and the job-table entry cleared.

Saving Output from Completed Jobs

If you reattach to a completed job, you can see the output of your job but the output is not saved for future reference. One way to save output of detached jobs is to use DOS redirection at the time you reattach a job. For example, assume an **on - spell** job has completed execution. It is listed in the job table as follows:

```
JOB    HOST        STATE       EXIT STATUS   COMMAND
[1]    d:host1     Done        exit(0)       spell memo
```

You can save the list of spelling errors when you reattach to the job by entering the command:

```
jobs %1 > typos
```

This command redirects the list from your screen into a file called **typos**.

You can also use pipes and redirection when you first issue the **on** command. See "Pipes and Redirection" on page 3-13.

Stopping Detached Jobs

You can halt detached jobs and clear them from the job table by reattaching to them, pressing **Ctrl-C** or **Ctrl-Break**, then responding with a to the resulting prompt.

You can also stop an AIX process by using the DOS **kill** command in either of these forms:

```
kill [-signal]  %jobnumber . . .
kill [-signal]    [host|driveletter:]  pid . . .
```

The *jobnumber* is the process number of the job in the job table, and *signal* is the AIX signal sent to the process. If you specify a *jobnumber*, the job-table entry is cleared automatically. If no job number is specified after **%**, then the lowest numbered job will be stopped.

If the second form is used, *pid* is the AIX process ID of the process to be signaled. If *pid* is negative, the entire AIX process group is signaled. If no *host* or *driveletter* is specified, the host associated with the current drive is used. If the current drive is not a virtual drive, an error message occurs. In addition, when the second form is used, the job-table entry is not cleared automatically.

The Access program **kill** command works similarly to the AIX **kill** command except that it accepts *%jobnumber* in addition to the process ID number argument. See **kill** in *AIX Operating System Commands Reference* for further information. The default signal sent is 15. A signal of 9 can be used to ensure stopping a process.

Notes:

1. Entering **kill -9** is not recommended unless other **kill** commands have failed, because the AIX program has no chance to perform cleanup operations before exiting.
2. Do not use **on** to execute an AIX **kill** command for detached jobs.

Pipes and Redirection

With the **on** command, you can use either DOS or AIX pipes and redirection symbols, or even combine the DOS and AIX versions of these mechanisms in a single command. You can accomplish nearly all useful operations using the DOS mechanisms. The following sections concentrate on DOS pipes and redirection. A few comments on AIX pipes and redirection are included in "Using Pipes and Redirection in Detached Tasks" on page 3-14.

DOS Pipes

You can use DOS pipes (|) with AIX programs that you invoke using **on**, as you would with DOS programs. The following examples show two different ways to accomplish the same process:

```
on - ls | find "chap"
on - ls | on - grep chap
```

Both versions locate and display all the file names in the current directory that contain the characters `chap`. The first example funnels the output of the AIX **ls** command through the DOS **find** command. The output of **ls** is returned to the DOS environment, and is then piped with the DOS pipe symbol through the DOS **find** program.

The second example pipes the output of **ls** to another command started by **on**. The output of **ls** is returned to the DOS environment by **on**, and is then rechanneled with the DOS pipe symbol to the **on** command, which then runs the AIX **grep** command.

DOS Output Redirection

You can redirect the output of **on**-initiated commands to specified files. DOS output redirection (>) allows you to direct the output to a file on any disk drive or directory where you have write permission. For example, to put the results of a command on a diskette, you would enter the following:

```
on - ls | find "chap" > a:tempfile
```

DOS redirection of output works because **on** sends the output back to your DOS environment. Thus, you can redirect the output of both AIX and DOS commands with the DOS output redirection mechanism (>).

DOS Input Redirection

You can use the DOS input redirection mechanism (<) to redirect standard input from a file on a local or virtual drive, or from the DOS keyboard to an **on** command.

Input redirection from a file is useful when you want to invoke an AIX command that operates on a file located on a DOS device, such as drive A. You can use input redirection to redirect the input of commands that read directly from standard input or a file. The **spell** command accepts input from standard input or a file. You can cause **spell** to take its input from a file called **names** by entering the following:

```
on - spell < a:names
```

The DOS input redirection mechanism (<) works together with the DOS drive designation `a:` to transfer the contents of the file **names** to the AIX **spell** command.

Note: You cannot use this form of redirection with AIX programs that cannot read from standard input.

You can also redirect the input for an AIX command from the DOS keyboard. For example, you can open a text file **temp** on the current drive and enter text into it as follows:

```
on - cat > temp < con:
This is line 1 of input.
This is line 2 of input.
```

To end the input and return to the DOS prompt, press **Ctrl-Z** and then press **Enter**. This sequence also closes the AIX standard input.

Using Pipes and Redirection in Detached Tasks

A detached **on** command that includes DOS pipes or redirection might not produce the desired results. This can happen because the DOS command processor, **command.com**, interprets the pipe and redirection symbols (<, >, and |), while **on** interprets other parts of the command line. This means that the job number and AIX process ID returned by the **on** command when you use an ampersand (&) will be piped or redirected, instead of the output from the AIX process.

If you want to redirect the output of a **spell** command into a file called **typos**, and you want to run **spell** as a detached process, you can use either of the following methods:

- Issue the command in the form:

  ```
  on - spell memo &
  ```

  The **on** command will display the job number and AIX process ID similar to the following:

  ```
  [1] 231
  ```

  (If you use a DOS output redirection symbol (>) and filename in front of the ampersand, you would save the job number and process ID, which is probably not what you want.)

  You can save the output from **spell** in the file **typos** when you reattach, as shown here:

  ```
  jobs %1 > typos
  ```

  The **on** command saves the output from an AIX process when it is run as a detached job, until you reattach to that job. Since the **aix2dos** function is built-in to **on**, redirecting the output when you reattach saves the file **typos** in DOS format.

- Use AIX output redirection rather than DOS output redirection in the **on** command.

  You use special symbols in the **on** command to perform AIX redirection and piping operations. The **on** utilities use a right brace (}) to indicate AIX output redirection, a left brace ({) to indicate AIX input redirection, and an exclamation point (!) to indicate an AIX pipe. These three symbols require spaces on both sides of them. For example:

  ```
  on - spell memo } typos &
  ```

  In this example, the output is not returned to the DOS environment, so it remains in AIX format rather than being converted to DOS format.

The following example shows how you can run **spell** as a detached job and use AIX piping and redirection to save **typos** in DOS format, without having to reattach.

```
on - spell memo ! aix2dos } typos &
```

This example uses an AIX pipe to send the **spell** output directly to the program **aix2dos** installed on the host, without returning to DOS or using a second **on** command. Because the output is not returned to the DOS environment, it remains in AIX format rather than being converted to DOS format. The program **aix2dos** on the AIX host is used to convert the list to DOS format, then AIX output redirection (}) puts the list into the file **typos**.

Notes:

1. To be properly interpreted by the **on** command, the special redirection and pipe characters ({, }, and !) require spaces surrounding them.
2. If you want **on** to interpret redirection or pipe symbols as regular characters, you must preceed the symbol with an AIX-style escape character (\). For example, to use **fgrep** to search the file **cprog.c** for any left braces, enter:

   ```
   on - fgrep "\{" cprog.c
   ```

   (The quotation marks are required by **fgrep** to indicate the characters to search for.)

For further information on using AIX pipes and redirection with **on**, see Chapter 7, "Command Summary."

Summary of Restrictions and Cautions

- The **on** command cannot run interactive AIX programs, except those that can be run with DOS console redirection as described in "DOS Output Redirection" on page 3-13 and "DOS Input Redirection" on page 3-13.
- AIX commands and shell scripts that contain an uppercase letter in their names can only be executed by using the prefix form of the **on** command.
- The **on** command cannot, in a single command line, initiate multiple AIX commands separated by semicolons.
- The **on** commands' use of the exclamation point and braces (as aliases for the AIX pipe and redirection symbols) works only when these symbols are preceded and followed by spaces.
- Although you can mix DOS and AIX pipes and redirection symbols in an **on** command, if you place a command containing any DOS redirection in the background of the AIX environment, the results can be unpredictable. Similarly, you cannot redirect DOS output to an AIX file. See "Using Pipes and Redirection in Detached Tasks" on page 3-14.
- The default 160-byte size of the DOS variable environment might not leave enough room to export all necessary variables to the AIX environment. See "Expanding DOS Environment Space" on page 3-8 for more information.
- The **on** utilities are not tested with, and might not run under, alternate DOS shell processors that completely replace **command.com**.

For more information, refer to Chapter 5, "Tailoring AIX Access for DOS Users," and Chapter 7, "Command Summary."

Chapter 4. Using Your Computer as an AIX Terminal

CONTENTS

About This Chapter 4-3
The Terminal Emulation Utility 4-4
Beginning a Terminal Emulation Session 4-4
  Communications Parameters 4-5
  Logging in over a LAN Connection Path 4-7
  Logging in over an RS-232 Connection Path 4-8
Switching between Terminal Emulation and DOS 4-9
Closing a Terminal Emulation Session 4-10
Using the Help Menu 4-11
The Personal Computer Keyboard 4-12
  The Standard Typewriter Area 4-12
  The Numeric Keypad 4-13
  The Function Keys 4-13
  PC Scancode Key Combinations 4-14
Send and Receive Files 4-15
  Using a Receive File 4-15
  Using a Send File 4-16

About This Chapter

This chapter tells you how to use terminal emulation, including:

- Beginning and ending a terminal emulation session.
- Switching between terminal emulation and DOS.
- Using the Help menu, status line, and keyboard.
- Transferring files.

Note: The procedures and examples described in this chapter assume that the Access program has been installed for your personal computer as described in Appendix A, "Installing AIX Access for DOS Users."

The Terminal Emulation Utility

The Access program terminal emulation utility allows you to communicate with an AIX host as if your personal computer were a VT100. Moreover, if the AIX host also runs DOS applications, you can communicate with those applications as if your personal computer were a PC Scancode terminal.

The terminal emulation utility automatically sets your personal computer to VT100 or PC Scancode terminal emulation, depending on whether you are communicating with an AIX or DOS process on the remote host.

Most of the attributes of a standard VT100 or PC Scancode terminal are supported by the utility, including the four programmable function keys, numeric keypad cursor control, paging, control and escape sequences, and flow control (**Ctrl-Q** and **Ctrl-S**). The 132-column mode is not supported.

All the PC Scancode terminal functions are supported, except for the **Left Shift-Alt-F9** combination, which has a special function and is described in "The Personal Computer Keyboard" on page 4-12.

Beginning a Terminal Emulation Session

You can conduct a terminal emulation session without establishing a host-file services session, or in conjunction with a file services session. However, it is important to note you cannot conduct both terminal emulation and host-file services sessions concurrently over an RS-232 connection path.

The following procedure starts a terminal emulation session. A terminal emulation session allows you to communicate directly with a host computer.

Note: If you already have a host-file services session in progress, change the default drive back to the drive that contains the Access program software and skip to step 3.

1. If you are using an Access program working diskette, insert the diskette in drive A and close the drive door. If you are using a fixed disk, leave the drive A door open.

   Note: If you are using a diskette, make sure your working diskette #1 is set up for the type of network interface adapter installed in your personal computer. The type of network interface adapter should be indicated on the diskette label.

2. Switch on your personal computer. If the system unit is already on, press and hold the **Ctrl** and **Alt** keys, then press the **Del** key.

   A message similar to the following is displayed:

   ```
   AIX Access for DOS Users Version 1.1 (Serial #10.1779) Initialized
   ```

   Note: Depending on other software that might be installed for your personal computer, you might see other messages as well.

   Following this message, the system executes the commands from the **autoexec.bat** file.

After the **autoexec.bat** commands execute, the system prompt is displayed. If you are using an Access program working diskette, the prompt is A>. If you are using a fixed disk the prompt is usually C>.

3. In response to the system prompt, type the following:

   em

   Press **Enter**.

   The terminal emulation logo screen is briefly displayed, with the message Initialization on the status line at the bottom of the screen. Then a host-selection menu, similar to the following, is displayed:

```
Select host:

     *zeus
      apollo
      athena
      com1
      com2




 1 2 3 4   Open session    local     F9 for help  ←→  REP
```

The Access program host menu screen shows you the connected AIX hosts. When your personal computer is equipped with one or more RS-232 ports, each port is identified by com*n*, where *n* is a number from 1 to 4 depending on the number of ports installed in your computer.

When there are more hosts than can be displayed on the screen, the Access program displays only one screen of hosts. Additional hosts are not displayed.

The **em** command can be issued from any drive, including a virtual drive under host file services. The **em** program can be in a directory specified in your search path, or you can include the drive and directory on the command line (for example, c:\aix\em).

Communications Parameters

The Access program allows you to change certain communications parameters when necessary. The Change parameter menu allows you to review and change these parameters.

You can change the communications parameters when you are establishing a terminal emulation session, or at any time during a session. Press the **F6** key to display the Change parameters menu.

The following shows the Change parameters menu for an RS-232 connection:

```
 Terminal mode:              Baud rate:

*No echo                     *9600
 Echo                         4800
 Local                        2400
 Scan code                    1200
                              600
                              300
                              150
                              110

                             Parity:

                             *Space
                              Mark
                              Even
                              Odd
                                      F6 to exit
  Tabs:
12345678901234567890123456789012345678901234567890123456789012345678901234567890
       T       T       T       T       T       T       T       T       T
  1 2 3 4    Change parameters       local           F9   for help  ←→  REP
```

The cursor will be positioned at the current setting for the first parameter, **Terminal mode**. The current setting for each parameter is indicated by an asterisk (*).

The following parameters appear on the Change parameters menu:

Terminal Mode Indicates what happens to characters you type at the keyboard. The **Terminal mode** can take any of the following options:

- **No Echo** ASCII codes for the characters you type are transmitted to the host, but the characters are not displayed on your screen until the host echoes them back to your personal computer. This is the default setting.
- **Echo** ASCII characters are displayed on your personal computer screen before being transmitted to the host. With this setting, you have no verification that the host actually receives the characters you type. However, the Echo setting should be used when the host communicates in the half-duplex mode.
- **Local** Characters are displayed on the screen, but they are not transmitted to the host.
- **Scan code** PC Scancodes for the characters you type are transmitted to the host.

Normally, when you log in to an AIX host for emulation, your terminal mode is set automatically to either No echo or Scan code, depending on what the current AIX process expects. Occasionally, the host process can end abnormally, leaving your terminal in one mode when the host requires another. In this situation, you can use the Change parameters menu to change your terminal mode.

Note: If you are in Scan code mode when this occurs, you must first press **Left Shift-Alt-F9** to invoke the Help menu, and then choose the Change parameters menu.

Baud Rate Indicates the data transmission rate, in bits per second, for data sent between your personal computer and the AIX host. The default setting is 9600 bits per second. This parameter is only displayed during an RS-232 terminal emulation session.

Parity Indicates the parity-checking scheme used for the data transmitted to and from your personal computer. The default is space parity. This parameter is only displayed during an RS-232 terminal emulation session.

Tabs Indicates where tabs are set during the terminal emulation session. The tab positions are indicated by the letter T on the line below the numbered ruler line. To set a tab, move the cursor to the line below the ruler line by using the **Cursor Down** key, use the **Cursor Left** and **Cursor Right** keys to place the cursor at the position for the new tab, and press the **Enter** key. The letter T is displayed. To remove a tab, place the cursor on the corresponding letter T, then press the **Enter** key. The letter T is no longer displayed.

When you are finished setting tabs, use the **Cursor Up** key to move the cursor to the ruler line, then press the **Tab** key to move back to the **Terminal mode** parameter.

To retain the current setting for a parameter on the Change parameters menu, press the **Tab** key to move the cursor to the next parameter field.

To change the setting for a parameter, first move the cursor to the parameter field you want to change by pressing the **Tab** key, then press the **Spacebar** to move the asterisk (*) to the next setting. When the asterisk is next to the desired setting, press **Enter**.

You can also use the cursor keys to move the cursor next to the desired setting in a parameter field, then press **Enter**. The asterisk remains at the current setting until you press the **Enter** key.

When you press the **Enter** key, the asterisk is left next to the new setting, and the cursor moves to the next parameter field. To move back to the previous parameter field, press and hold the **Shift** key and press the **Tab** key.

When all of the parameter fields are set as desired, press the **F6** key to save the values and leave the Change parameters menu. (Alternately, you can press the **Enter** key when the cursor is next to the `F6 to exit` field.)

Logging in over a LAN Connection Path

Use the following procedure to establish a terminal emulation session with an AIX host connected through a LAN connection path.

1. At the host-selection menu, press the **Spacebar** until the asterisk (*) is next to the name of the host you want to connect to, and then press **Enter**.

   The following prompt is displayed to the right of the host-selection list:

   ```
   Extra pages (0-4) [4]:
   ```

   When you exit a terminal emulation session, the **em** program always saves the last screen (or page) of the session in the file **em.ses**. You can save up to four additional pages by entering the number of aditional pages at this prompt. The default is four additional pages. Each page requires about 4KB of disk storage space.

   The file **em.ses** is normally created in the current directory on the current drive. You can set the DOS environment variable **EMSES** to a specific drive and directory, and the file **em.ses** is saved in that directory.

   The screens saved in the file **em.ses** are used so that when you return to a terminal emulation session from a DOS session, the screen looks just as it did when you left it. Refer to "Switching between Terminal Emulation and DOS" on page 4-9 to see how to exit from your terminal emulation session, and how to return to it. For information on

setting the **EMSES** environment variable, see Chapter 5, "Tailoring AIX Access for DOS Users."

You may return to the host-selection menu at this point by pressing **Esc**.

2. To accept the default number of pages, press **Enter**. Otherwise, type the number of pages you want to have saved and press **Enter**.

   The following message is displayed below the extra-pages prompt:

   ```
   Opening the connection . . .
   ```

   If the connection cannot be established, you will see an error message. Verify that the host you selected is the correct one, and try the procedure again. If you still cannot establish a connection, press the **F10** key to return to DOS and contact the person who administers your system.

   When the LAN connection is successfully established, the screen clears and the AIX host login banner and prompt appear. If the screen remains blank or contains erroneous characters, your personal computer terminal mode may differ from the one the host is currently expecting. Access the Change parameters menu by pressing **F6**, and change your terminal mode to `No echo`. If the mode is already set to `No echo`, change it to `Scan code`. If you cannot access the Change parameters menu, or if you cannot get the AIX user-name prompt, press the **F10** key to return to DOS and contact the person who admisters your system.

3. Complete the login process by entering your AIX user name and password.

Note: AIX determines terminal characteristics by referring to the **TERM** environment variable. If you use the Access program terminal emulator regularly, you should include the following lines in the file **.profile** in your AIX home directory on your AIX host to set the **TERM** environment variable:

```
TERM=VT100-em
export TERM
```

If you use the C shell, add the following lines in your **.login** file on your AIX host:

```
setenv TERM=VT100-em
export TERM
```

Note: If your AIX host is an RT, use `TERM=VT100` instead of `TERM=VT100-em`.

Logging in over an RS-232 Connection Path

Use the following procedure to begin a terminal emulation session over an RS-232 connection path.

1. At the host-selection menu, press the **Spacebar** until the asterisk (*) is next to the RS-232 port that you want to connect through, and then press **Enter**.

   The following prompt is displayed to the right of the host-selection list:

   ```
   Extra pages (0-4) [4]:
   ```

   When you exit a terminal emulation session, the **em** program always saves the last screen (or page) of the session in the file **em.ses**. You can save up to four additional pages by entering the number of aditional pages at this prompt. The default is four additional pages. Each page requires about 4KB of disk storage space.

   The file **em.ses** is normally created in the current directory on the current drive. You can set the DOS environment variable **EMSES** to a specific drive and directory to have the file **em.ses** saved in that directory.

   The screens saved in the file **em.ses** are used so that when you return to a terminal emulation session from a DOS session, the screen looks just as it did when you left it. Refer to "Switching between Terminal Emulation and DOS" on page 4-9 to see how to exit from your terminal emulation session, and how to return to it. For information on

setting the **EMSES** environment variable, see Chapter 5, "Tailoring AIX Access for DOS Users."

You may return to the host-selection menu at this point by pressing **Esc**.

2. To accept the default number of pages, press **Enter**. Otherwise, type the number of pages you want to have saved and press **Enter**.

   The following message is displayed below the extra-pages prompt:

   ```
   Opening the connection . . .
   ```

   If the connection cannot be established, you will see an error message. Verify that the RS-232 port you selected is the correct one, and try the procedure again. If you still cannot establish a connection, press the **F10** key to return to DOS and contact the person who administers your system.

   If a connection is established, the Access program displays the Change parameters menu to let you set the communications parameters.

3. Set any necessary parameters as described in the section "Communications Parameters" on page 4-5 and press **F6**.

   Note: If you are using modems to communicate with your host, set the baud rate to match the baud rate of your modems. After you press **F6** to leave the Change parameters menu, you can make the connection as described in Appendix C, "Using a Dial-Up Modem."

   The AIX host banner and login prompt appear.

   If the prompt does not appear, press **Enter** again. If the prompt still does not appear, then the baud rate of the host probably does not match the baud rate you selected. (You may see erroneous characters on the screen at this time.) To change the host baud rate, hold down the **Ctrl** key and press the **Break** key repeatedly.

   If the login banner still does not appear, your personal computer terminal mode may differ from the one the host is currently expecting. Try pressing **F6** to return to the Change parameters menu and change your terminal mode to `No echo`. If the mode is already set to `No echo`, change it to `Scan code`.

4. Complete the login process by entering your AIX user name and password.

Note: AIX determines terminal characteristics by referring to the **TERM** environment variable. If you use the Access program terminal emulator regularly, include the following lines in the file **.profile** in your home directory on your AIX host to set the **TERM** environment variable:

```
TERM=VT100-em
export TERM
```

If you use the C shell, add the following lines in your **.login** file on your AIX host:

```
setenv TERM=VT100-em
export TERM
```

If your AIX host is an RT, use `TERM=VT100` instead of `TERM=VT100-em`.

Switching between Terminal Emulation and DOS

The Access program allows you to switch between your terminal emulation session and a DOS session. Press the **F10** key from the terminal emulation session and you exit to DOS. You can use your personal computer to perform any common DOS tasks, as well as to establish and use the Access program file services sessions.

Note: If your personal computer is in the PC Scancode mode, you must press **Left Shift-Alt-F9** to display the Help menu before pressing **F10**.

The Access program terminal emulator saves your last few terminal emulation session screens in the file **em.ses**. Normally **em.ses** is saved in the directory where you first executed the **em** command. You can tailor the Access program to save **em.ses** in a specific directory, regardless of the current drive and directory (see Chapter 5, "Tailoring AIX Access for DOS Users").

To return to your terminal emulation session, change back to the same directory and drive from which you executed the previous **em** command and execute em again.

Your terminal emulation session screen is displayed as it looked when you pressed the **F10** key. Any **cd** commands issued while you are in DOS will not affect the current directory for your terminal emulation session.

Closing a Terminal Emulation Session

Use the following procedure to end your terminal emulation session.

1. Switch to your terminal emulation session and log off of your AIX host by entering:

   ```
   logout
   ```

   The host login prompt is displayed.

2. Press the **F7** key. A Close session menu similar to the following is displayed:

   ```
                    * Close COM1:
                      Leave Open


   1 2 3 4  Close session        local       F9 for help   ←→  REP
   ```

   The **Close** option will specify the host name or RS-232 port. If necessary, press the **Spacebar** to move the asterisk to the **Close** option.

3. Press **Enter** and the Access program terminal emulator presents the host-selection menu, giving you an opportunity to establish another terminal emulation session.

4. Press the **F10** key to return to DOS.

   If you always log in to the same host, you can skip steps 2 and 3 in the preceding procedure. This causes the next **em** command to bypass the host-selection menu and display the login prompt of the same host.

Using the Help Menu

The Access program terminal emulation provides a Help menu that shows the meanings of the function keys, the current host connection path, instructions for making menu selections, and a description of the status line, as follows:

```
   F5  - Open session              F6   - Change parameters
   F7  - Close session             F9   - Help toggle
 *  F10 - Exit to DOS               AF6  - Local toggle

   AF9 - Open file           AF10 - Close file

            Open sessions:  *COM1:

To select from a menu:

  Space moves to next item
  Return selects item
  Tab moves to next section
  Shift/Tab moves to previous

  or use arrow keys to move cursor
  to an item, then press Return

            Current                                          Caps Num  Ins-
  LED'S     session    File      Mode                        lock lock  ert
 1 2 3 4     Menu               local  F9  for help               ←→  REP
```

You can display the Help menu at any time during your terminal emulation session by pressing the **F9** key if you are running an AIX process, or **Left Shift-Alt-F9** if you are running a DOS process on the AIX host.

The Help menu contains four areas:

- Function key selection
- Current host or RS-232 port
- Instructions
- Status.

An asterisk (*) indicates the currently selected function key. The current host or RS-232 port is specified by the **Open sessions** parameter. Following the **Open sessions** parameter are instructions describing how to select items.

The last line on your screen shows the status of the terminal emulation session. The status line is divided into three sections:

- An LED status section
- A session status section
- A keyboard status section.

The LED status section is displayed at the leftmost edge of the status line. It consists of the numbers 1, 2, 3, and 4, which represent the LEDs on a standard VT100 keyboard.

The session status section is displayed in a reverse video band at the center of the screen.

The keyboard status section on the right shows the status of the **Caps Lock**, **Num Lock**, and **Insert** keys. When the **Caps Lock** key is on, CAPS is displayed slightly to the right of the session status (the reverse video band). When the **Caps Lock** key is off, this section of the status line is blank. When the **Num Lock** key is on, NUM is displayed to the right of the CAPS indicator. When the **Num Lock** key is off, a pair of left and right arrows replaces NUM. REP in the rightmost field of the status line indicates the normal replace mode. When you use the **Insert** key in local mode, INS is displayed there instead.

You can use the Help menu to review the function key choices and then execute one of these functions. To select a function key for execution, press the **Spacebar** until the asterisk is opposite the function you want to execute, then press the **Enter** key. Alternately, simply press the function key you want.

To return to your terminal emulation session, select the **F9** function key or simply press **F9**.

The Personal Computer Keyboard

When you are using the Access program terminal emulator for VT100 emulation, most of the keys in the standard typewriter portion of your keyboard send the expected ASCII character sequences. However, a few of these keys have special meanings, as do the function keys and some of the keys in the numeric keypad. Some keys have no function at all when you use the terminal emulator, while others have special functions when used in combinations.

The following sections describe these keys and key combinations for VT100 emulation.

The Standard Typewriter Area

Caps Lock The **Caps Lock** key locks alphabetic characters in uppercase. If you press the **Shift** key while in **Caps Lock**, the alphabetic characters have their unshifted or lowercase functions.

Note: Characters other than alphabetics are not affected by **Caps Lock** and require using the **Shift** key to obtain the shifted character.

Enter or ↵
The **Enter** key sends the ASCII carriage return character (octal 015).

Print Screen or PrtSc
When you enter terminal emulation from a file services session, pressing the **Print Screen** key (**Shift-PrtSc** for PCs, ATs, and XTs), sends the text on your screen to either your local printer or the remote AIX printer, depending on which printer you specified for the LPT1 print stream with the the Access program **printer** command.

When you enter terminal emulation without first establishing a file services session, pressing the **Print Screen** key (**Shift-PrtSc** for PCs, ATs, and XTs), sends printing to the local printer attached to your personal computer.

Note: If printing is directed to a local printer and there is no printer attached to your personal computer, your terminal might become locked for a short time.

Backspace or ←
The **Backspace** key moves the cursor one character to the left, deleting the current character. The **Backspace** key transmits the ASCII backspace character (octal 010).

Alt The **Alt** key alone has no function in terminal emulation but is used in combination with other function keys to perform certain functions. These combinations are described in "The Function Keys" on page 4-13.

Ctrl The **Ctrl** key alone has no function in terminal emulation but can be used with other keys to transmit control characters to the AIX host.

Ctrl-Q You can start the flow of characters by pressing the **Ctrl** and **Q** keys in combination. Use **Ctrl-Q** after you use **Ctrl-S** to stop the flow of characters.

Ctrl-S You can stop the flow of characters by pressing the **Ctrl** and **S** keys in combination.

Esc The **Esc** key transmits the ASCII escape character (octal 033) to the AIX host.

Tab The **Tab** key transmits the ASCII tab character (octal 011).

Shift or ↑ Pressing either **Shift** key in combination with one of the character keys in the typewriter area causes the uppercase character to be displayed and transmitted.

Note: The **Caps Lock** key reverses the function of the **Shift** keys for alphabetic characters.

The Numeric Keypad

Num Lock Pressing the **Num Lock** key places the numeric keypad in numeric mode. Pressing the **Num Lock** key again returns the keypad to cursor control mode.

Ctrl-Scroll Lock
The **Ctrl** and **Scroll Lock** keys, in combination, send a break signal over an RS-232 connection path. This is used to change the baud rate on an AIX host that supports automatic baud rate setting. The key combination has no meaning when the terminal emulation session is over a LAN.

Insert or Ins
The **Insert** key allows you to insert text in existing text on your display when the emulator is in local mode. Text inserted in local mode is not transmitted to the AIX host. The **Insert** key can also have special meaning to some AIX applications.

Delete or Del
The **Delete** key transmits the ASCII delete character (octal 177) to the AIX host.

↑↓←→ The cursor control keys move the cursor up or down one line, and left or right one character. The IBM PS/2 keyboard provides two sets of cursor control keys.

Home The **Home** key moves the cursor to the top left corner of your display.

End The **End** key moves the cursor to the right of the last character on your display.

Ctrl-PgUp The **Ctrl** and **Page Up** keys, in combination, are used to display text that has scrolled off the top of your screen during your terminal emulation session. Each time you press the **Ctrl** and **Page Up** keys, the previous page of text (25 lines) is displayed, up to the number of pages you specified to be saved when you opened the terminal emulation session.

Ctrl-PgDn The **Ctrl** and **Page Down** keys, in combination, are used to display text that scrolls off the bottom of your screen when you use the **Ctrl** and **Page Up** keys. Each time you press the **Ctrl** and **Page Down** keys, the next page of text (25 lines) is displayed, until the current session screen is displayed.

The Function Keys

F1 The **F1** key transmits the three-character ASCII sequence **Esc**, O, and P (octal 033, 117, and 120). The effect of this sequence is application-dependent and is the same as the **PF1** key on a VT100 terminal.

F2 The **F2** key transmits the three-character ASCII sequence **Esc**, O, and Q (octal 033, 117, and 121). The effect of this sequence is application-dependent and is the same as the **PF2** key on a VT100 terminal.

F3 The **F3** key transmits the three-character ASCII sequence **Esc**, O, and R (octal 033, 117, and 122). The effect of this sequence is application-dependent and is the same as the **PF3** key on a VT100 terminal.

F4 The **F4** key transmits the three-character ASCII sequence **Esc**, O, and S (octal 033, 117, and 123). The effect of this sequence is application-dependent and is the same as the **PF4** key on a VT100 terminal.

F5 For efficiency, the Access program automatically issues an **F5** to present you with the host-selection menu when you invoke the terminal emulator or close a terminal emulation session. You should not use the **F5** key when you have a terminal emulation session open.

F6 Pressing the **F6** key displays the Change parameters menu.

When you begin a terminal emulation session over an RS-232 connection path, you automatically see the Change parameters menu. You can also invoke it at any time during a terminal emulation session to change any parameters.

After you begin a terminal emulation session over a LAN connection path, you can press **F6** to display the Change parameters menu at any time during your terminal emulation session, but you can only change the terminal mode and tab settings.

F7 Pressing the **F7** key displays the Close session menu.

F8 The **F8** key has no meaning for terminal emulation to an AIX host.

F9 Pressing the **F9** key displays the Help menu.

Note: In PC Scancode mode, you must use **Left Shift-Alt-F9** to display the Help menu.

F10 Pressing the **F10** key suspends the terminal emulation session and returns you to DOS.

F11 The **F11** key on the IBM PS/2-style keyboard has no meaning for terminal emulation to an AIX host.

F12 The **F12** key on the IBM PS/2-style keyboard has no meaning for terminal emulation to an AIX host.

Alt-F6 Pressing the **Alt** and **F6** keys in combination changes the emulator to local mode. In local mode, characters you type are displayed on your screen but not transmitted to the host. Pressing **Alt** and **F6** again returns you to remote mode.

Alt-F9 Pressing the **Alt** and **F9** keys displays the Open file menu. You can transfer files between the host and your personal computer in either direction, as described in "Send and Receive Files" on page 4-15.

Alt-F10 Pressing the **Alt** and **F10** keys closes files opened during file transfer.

Ctrl-Alt-Del
Pressing the **Ctrl**, **Alt** and **Del** keys in combination restarts your personal computer.

PC Scancode Key Combinations

When you use the Access program terminal emulation in PC Scancode mode, all the keys on the keyboard send the expected scancode character sequences with the exception of the **Left Shift-Alt-F9** and **Ctrl-Alt-Del** combinations. These combinations have the following meanings in PC Scancode mode:

Left Shift-Alt-F9
Pressing the **Left Shift**, **Alt** and **F9** keys in combination displays the Help menu. You can choose any of the function key functions, including **F6** to display the Change parameters menu or **F10** to exit to host file services or local DOS.

Ctrl-Alt-Del

Pressing the **Ctrl**, **Alt** and **Del** keys in combination restarts your personal computer. This key combination cannot be used to switch the AIX host from a DOS process to an AIX process. To switch, you must first exit the DOS process by typing `quit`.

Send and Receive Files

All interaction during a terminal emulation session may be stored in a ***receive*** file for later review. This allows you to log on to a host, obtain the desired information, save the information in a receive file, and log off of the host. You can review the information at a later time, thereby minimizing the connect-time to the host.

When the sequence of interactive commands and input are known in advance, a ***send*** file may be used as input to the host during a terminal emulation session. A send file contains all of the keystrokes that you would normally type during a session.

Send and receive files may be used simultaneously to conduct a session automatically. The send file must contain all of the commands and other interactive input required by any application. The host will automatically log off when the end of the send file is reached.

Notes:

1. It is easier, faster, and more reliable to transfer files between your personal computer and the AIX host by copying to or from a virtual drive in a host file services session.
2. You can send and receive only ASCII text files. Binary files cannot be used as send and receive files.

Using a Receive File

The following procedure can be used to store all of the text displayed during a terminal emulation session into a receive file.

1. After logging into the host, press **Alt-F9**.

   A menu similar to the following is displayed:

```
        Select transfer direction:

                * Receive file
                  Send file


1 2 3 4   Open file      local       F9 for help   ←→ REP
```

2. Press **Enter** to choose the `Receive file` option.

   The following prompt is displayed on the menu below the `Send file` option:

   ```
   File name:
   ```

3. Type the name of a new file that you want to receive the displayed data. (This is the *receive* file.) You can also include a drive specifier. Press **Enter**.

   The terminal emulator opens the new file on the drive you specified. If you did not specify a drive, the file is opened on the current drive (the drive from which you entered terminal emulation).

   The menu screen clears and your Terminal Emulation Session screen is redisplayed.

4. Continue with your terminal emulation session. The commands you entered and the results of the commands are displayed on your screen and sent to the receive file on your personal computer or the virtual drive.
5. To end the file transfer, press **Alt-F10**. The Access program closes the file. Any data displayed after closing the file will not be saved.

Using a Send File

The following procedure can be used to send commands and data stored in a send file to a host during a terminal emulation session.

1. Create an ASCII file using any text editor. This file may contain AIX commands or other input required to complete a terminal emulation session. If any interactive commands are contained in the file, all input required by the command must also be in the file in the same order as you would normally type it.
2. Log in to the host. (Follow the procedure for using a receive file if that is also desired.)
3. Press **Alt-F9**.

   A menu similar to the following is displayed:

```
          Select transfer direction:

                  * Receive file
                    Send file


 1 2 3 4   Open file              F9 for help       ←→  REP
```

4. Press the **Spacebar** to move the asterisk (*) and choose the `Send file` option. Press **Enter**.
5. The following prompt is displayed on the menu below the `Send file` option:

   `File name:`
6. Type the name of the send file that you created in step 1. You can also include a drive specifier. Press **Enter**.

   The terminal emulator looks for and opens the file on the drive you specified. If you did not specify a drive, the terminal emulator searches for the file on the current drive (the drive from which you entered terminal emulation).

   The following prompt is displayed on the menu below the `file name` option:

```
Select sending speed:

    * Character at a time
      Line at a time
      Whole file
```

7. Press the **Spacebar** until the asterisk (*) is opposite the desired option. Press **Enter**.

   The menu screen clears and your terminal emulation session screen is redisplayed.

   Note: Normally, you should use `Whole file` option. You may want to try the other options to determine which method works best for your system.

8. When the end of the file is reached, the Access program closes the file and automatically logs off of the host.

   Note: If a receive file is also being used, remember to close it.

Chapter 5. Tailoring AIX Access for DOS Users

CONTENTS

About This Chapter 5-3
AIX Access for DOS Users Program Files 5-4
Putting AIX Access for DOS Users in a Subdirectory 5-5
Putting AIX Access for DOS Users Files on a Virtual Drive 5-6
Putting DOS Applications on a Virtual Drive 5-6
Customizing File Services 5-7
Setting Search Paths 5-7
Setting a Default Search Path 5-8
Changing the Maximum Number of Virtual Drives 5-8
Changing the Maximum Number of Open Files under DOS 5-8
Changing the Job Table Size 5-9
Setting the Printer Environment 5-9
Setting the on Command Variables 5-9
The ONNAME Environment Variable 5-10
The ONPREFIX Environment Variable 5-10
Setting the Terminal Emulation Environment Variable 5-11
Using Terminal Emulation to an AIX/RT Host 5-11

About This Chapter

This chapter explains how to tailor your installation to put the Access program where it is most convenient for your use. It also describes how to create directories on the virtual drive for your DOS applications and how to access public directories to share DOS applications with other users.

In addition, this chapter describes how to change certain configuration options and how to set certain environment variables that allow you to establish the way you want the Access program to perform particular functions.

AIX Access for DOS Users Program Files

The Access program comprises a number of files that are installed on your working diskettes or fixed disk when you choose a standard installation according to the instructions in Appendix A, "Installing AIX Access for DOS Users." Use these files to initialize the Access program, establish a connection to your AIX host, and provide additional utilities that extend the power of the Access program.

Figure 5-1 lists the Access program files.

Files Required to Initialize Access Program

autoexec.bat	Invokes **pciinit** automatically when DOS is loaded.
bridge.sys	Provides basic Access program support.
config.sys	Identifies device driver to be loaded by DOS.
l*xxx***.sys**	LAN device driver for a personal computer bus. (*xxx* indicates the driver for your network interface adapter.)
m*xxx***.sys**	LAN device driver for a microchannel bus personal computer. (*xxx* indicates the driver for your network interface adapter.)
pciinit.exe	Initializes the Access program.

Files Required for Host-File Services

login.exe	Establishes a host connection for host file services.
logout.exe	Terminates a host-file services session.

Files Required for Terminal Emulation

em.com	Establishes a terminal emulation session.
nty.exe	Initializes terminal emulation for LAN connections to an AIX IBM PS/2 host computer or an AIX/370 host computer.

Access Program Utilities

aix2dos.exe	Converts text files in AIX format to DOS format.
dos2aix.exe	Converts text files in DOS format to AIX format.
exrecovr.exe	Recovers **vi** files after a DOS system failure.
jobs.exe	Displays the status of remote AIX jobs initiated with the **on** command.
kill.exe	Kills remote AIX jobs initiated with the **on** command.
on.exe	Executes AIX commands on a specified remote host.
printer.exe	Directs output to local or remote printer.
udir.exe	Lists virtual drive directory contents in AIX style.
vi.exe	PC version of the AIX **vi** editor.
doswhat.exe	Displays version information for the Access program executable files.
setdebug.exe	Controls DOS system-call logging.

Access Program Development Utilities

pcilcd.lib	Small code, large memory model for Lattice C.
pcilcl.lib	Large memory model for Lattice C.
pcilcp.lib	Large code, small data model for Lattice C.
pcilcs.lib	Small memory model for Lattice C.
pcimscc.lib	Compact memory model for Microsoft C.
pcimscs.lib	Small memory model for Microsoft C.

Figure 5-1 (Part 1 of 2). Access Program Files

pcimscm.lib	Medium memory model for Microsoft C.
pcimscl.lib	Large memory model for Microsoft C.
ipc.h	Interprocess Communications header file.
memmdl.h	Manifest constants header file.
pcilib.h	External PCILIB function call return types header file.

Figure 5-1 (Part 2 of 2). Access Program Files

Putting AIX Access for DOS Users in a Subdirectory

When you install the Access program on your personal computer with a fixed disk according to the instructions in Appendix A, "Installing AIX Access for DOS Users," you can choose to install all the Access program files in the root directory or to install some of them in a subdirectory. If you choose to install all the files in the root directory and later decide it would be more convenient to work from a subdirectory, you can relocate all the Access program files, except **autoexec.bat** and **config.sys**, to a subdirectory. These two files must remain in your root directory.

Use the following procedure to relocate the Access program files to a subdirectory:

1. Change your working directory to the root directory by entering:

   ```
   cd \
   ```

2. Create the subdirectory:

   ```
   mkdir pcibin
   ```

3. Copy all the Access program files, except **autoexec.bat** and **config.sys** to the new subdirectory one at a time. For example:

   ```
   copy dos2aix.exe pcibin
   ```

4. Edit your **autoexec.bat** file, adding the following line before the line containing the **pciinit** command:

   ```
   path=c:\pcibin
   ```

   Note: If you already have a **path** command in your **autoexec.bat** file, add **c:\pcibin** to your existing **path** command rather than adding a new **path** command.

   Also, note that the `c:` in this example should be replaced by the drive specifier for the directory in which you have placed the Access program files.

5. Modify the **config.sys** file to insert the appropriate path qualifier before each **device** command. For example, change:

   ```
   device=lxxx.sys
   ```

   (where *xxx* is a mnemonic for the driver for your network interface board) to:

   ```
   device=pcibin\lxxx.sys
   ```

6. Erase the original Access program files from the root directory. For example:

   ```
   erase dos2aix.exe
   ```

Putting AIX Access for DOS Users Files on a Virtual Drive

You need to keep only the files required to initialize the Access program and the files required for host file services on your local drive. The remaining files can be executed from the virtual drive once you have established a host-file services session. In addition, you can put the **logout** program on the virtual drive.

If you want to establish terminal emulation sessions without first establishing a host-file services session, you must also keep the **em.com** file on your local drive.

Use the following procedure to install the Access program utilities, and any other executable files you want on the virtual drive, in your home directory:

1. Initialize the Access program and establish a host-file services session.
2. Change your current drive to the virtual drive by entering:

   ```
   d:
   ```

   (Substitute the drive specification of the virtual drive you want to use.)
3. Be sure your home directory is the current directory on the virtual drive by entering:

   ```
   cd
   ```

   Verify that the directory displayed is your home directory (or the directory in which you want to install the Access program files). If the current directory is not the one you want, use the **cd** command to change to the appropriate directory.
4. Change back to your local drive by entering:

   ```
   a:
   ```

   (or `c:` if your personal computer has a fixed disk.)
5. Copy the Access program utility files one by one to the virtual drive. For example:

   ```
   copy  dos2aix.exe  d:
   ```
6. Erase the files you copied from your Access program working diskette. (You can restore them at any time by copying them back from the virtual drive or from your backup diskettes.) Make sure that you erase only the Access program utility files. Do not erase any files required to initialize the Access program or required for host file services.

You may want to consolidate frequently used commands or programs in a central, easily accessible directory. Since such directories usually contain only executable binary files, they are sometimes called **bin** directories (short for binary).

You can create a personal **bin** directory on your virtual drive and copy the Access program utilities to that directory instead of your home directory, using the same procedure described above. To create a **bin** directory, use the DOS command **mkdir** to create a directory called **pcibin**.

Be sure your search path includes these directories so DOS can find your Access program files.

Putting DOS Applications on a Virtual Drive

You can also use personal **bin** directories on a virtual drive to contain your DOS applications. To install DOS applications on a virtual drive, first establish a host-file services session, create a **bin** directory for the application, and change to that directory. Then, follow the manufacturer's instructions for installing the DOS application on a fixed disk.

Some DOS applications create or append to an existing **autoexec.bat** file in the **root** directory to invoke the application automatically when you turn on your personal computer.

When you install such an application on the virtual drive, this file is created on the virtual drive. However, since you do not have a connection to the virtual drive when you turn on your personal computer, the **autoexec.bat** file cannot start the application. You should rename the **autoexec.bat** file to a unique name that suggests the application and has a **.bat** extension (for example, **startws.bat**). Whenever you want to invoke the application, change your current directory to the **bin** directory in which you installed the application and type the file name without the extension. For example:

```
startws
```

Copy-protected DOS applications that require a key diskette in drive A before they can be started from a fixed disk require the same key diskette when you invoke them from the virtual drive. Installing DOS applications on the virtual drive does not circumvent any copy-protection mechanism that might exist for the application.

You are responsible for observing the licensing restrictions associated with any DOS applications you install on the virtual drive. Unless the licensing agreement permits you to share the software, you should only install applications in your own directories, while removing execute permission for all other users from those directories.

Public-domain software and DOS applications for which you have a multi-user licensing agreement can be installed in public directories on the virtual drive. Generally, the system administrator installs these packages and can tell you what applications are available at your installation and where they are.

Be sure your search path includes these directories so DOS can find the applications.

Customizing File Services

The following sections describe how to tailor the Access program for a custom file services session.

Setting Search Paths

In order to access the Access program utilities or DOS applications stored on the virtual drive, you have to tell DOS where to find them. You can do this each time you execute a command by first changing to the drive and directory that contain that command.

An easier way to tell DOS where to find commands is to set a ***search path***. A path is a list of directory locations that you want DOS to search to find a command or program. You can specify a search path at any time using the DOS **path** command. The search path you set stays in effect until you issue another **path** command.

There are two very important things to remember when you specify directories on the virtual drive in your **path** command:

- You must have a host-file services session in progress for DOS to search directories on the virtual drive.
- When you log off of a host-file services session, but plan to continue using local DOS, you should reset your search path to exclude any references to the virtual drive.

If you relocated the Access program files to a subdirectory on your personal computer fixed disk, be sure you include the subdirectory any time you change the search path.

Setting a Default Search Path

To avoid having to remember to set your search path to include the virtual drive, you can include the **path** command in your **autoexec.bat** file, specifying the virtual drive identifier the Access program normally returns to you at login.

However, setting **path** in the **autoexec.bat** has some disadvantages. Whenever the virtual drive specifier changes, it prevents you from using your Access program diskette until you reset the entire search path. It also forces you to reset the search path whenever you want to use DOS without establishing a host-file services session.

When DOS uses the search path and encounters a drive specification for which it does not have access (for example, the virtual drive when no host-file services session is in progress), DOS returns an `invalid drive` message and stops searching for the command or program you requested.

You can prevent this by creating a default search path that is invoked only after you establish a host-file services session. To create a default search path, use the following procedure:

1. Establish a host-file services session.
2. In your home directory on the virtual drive, create a file called **setpath.bat**, which contains a **path** command in the following form:

   ```
   path=%1:\usr\pcibin;a:\
   ```

   Include all the directories you want DOS to search the order in that you want them searched.

You can set the default search path as soon as you establish a host-file services session by changing to the virtual drive, then typing:

```
setpath d
```

where *d* is the virtual drive identifier. DOS substitutes this identifier for every %1 in the **path** command included in **setpath.bat**, then executes the command.

Changing the Maximum Number of Virtual Drives

By default, the Access program is automatically configured to support a maximum of four virtual drives. If you want to be able to establish connections to more than four virtual drives, you can change this configuration option.

Using any editor, append /d:*nn* to the `device=bridge.sys` line in your **config.sys**. file, where *nn* is the maximum number of virtual drives, up to 16. For example, if you want to establish connections to as many as 10 virtual drives, modify the **device** command in **config.sys** to read:

```
device=bridge.sys /d:10
```

This change takes effect when you restart your personal computer.

Changing the Maximum Number of Open Files under DOS

In your **config.sys** file, you can change the number of open files on your local drive using the DOS **files** command. However, this value does not change the maximum number of open files allowed on virtual drives. On a virtual drives, you can have up to 20 open files. Of these 20, three are used by standard input, standard output, and standard error.

Changing the Job Table Size

When you bring up your personal computer to start the Access program, a job table is initialized with 10 entries. For most users, a 10-entry job table is sufficient.

If you want to configure a job table with a larger or smaller number of job entries, modify the following line in the **config.sys** file:

```
device=bridge.sys /j:nn
```

where *nn* is an integer from 1 to 30 representing the number of entries in the job table to be configured.

The job table is configured to the new size the next time your personal computer is restarted.

The job table holds one entry for each task initiated with **on**. However, remember that one AIX task can generate several child processes. For example, a single **spell** command can appear in the DOS job table as one running process but would show up in an AIX **ps** list as four or five processes.

All AIX computers have a limit on the number of processes each user can have running at one time. This limit varies from system to system, but falls in the range of 20 to 30 processes per user. If each **on**-initiated AIX task generates several child processes, the limit on total number of processes is easily reached.

Setting the Printer Environment

The Access program allows you to direct your remote printing requests to any one of three remote print streams:

- LPT1
- LPT2
- LPT3

You can associate a particular AIX command string with each stream, using the **printer** command.

To establish your own default environment for each print stream, include **printer** commands in your **autoexec.bat** file. The print command and options you specify for each print stream are automatically in effect when you begin your host-file services session. They remain in effect until you explicitly change them by entering a new **printer** command for that print stream during the session. For more information, see "Printing with AIX Access for DOS Users" on page 2-17.

In addition, you can specify a default AIX command string that the Access program uses whenever no other print command is in effect for a print stream by including the following line in your **autoexec.bat** file:

```
set printprog=printprogram
```

where *printprogram* is the default AIX command string for all undefined print streams. This default overrides the system default set by the system administrator.

Setting the on Command Variables

The next two sections discuss setting the **ONNAME** and **ONPREFIX** environment variables to further customize your file services session.

The ONNAME Environment Variable

The **on** command recognizes the DOS environment variable **ONNAME**, which is provided to help resolve file name conflicts.

Since there can be several copies of **on**, each with a different name, **on** uses the contents of the **ONNAME** variable to recognize the default name for itself. If no **ONNAME** variable is specified, the value **on** is presumed. This is how **on** distinguishes between the commands **on -ls** and **ls**, when you have made a copy of **on.exe** named **ls.exe**.

If you want to rename **on**, you must specify the new name in the **ONNAME** variable. You can also rename **kill.exe** and **jobs.exe** in this manner.

The **ONNAME** variable recognizes three comma-separated fields representing the internal names of **on.exe**, **kill.exe** and **jobs.exe**, respectively. By default, the three fields have the values **on**, **kill**, and **jobs**. Any field can be blank to retain the default name, and trailing commas can be omitted.

For example, suppose you want to type `run - ls -l` instead of `on - ls -l`. Follow these steps:

1. Rename **on.exe** to **run.exe**.
2. Put the following line in your **autoexec.bat** file:

```
set onname=run
```

If you want to rename **jobs.exe** to **j.exe** at the same time, use the **set** command:

```
set onname=run,,j
```

If you simply rename **on.exe**, **kill.exe**, or **jobs.exe** without also changing the **ONNAME** variable, **on** attempts to run an AIX command with the specified name.

The parameters specified in the **ONNAME** variable must be in lowercase letters. In a **set ONNAME** command, every letter to the right of the equal sign must be lowercase.

The ONPREFIX Environment Variable

The **on** command executes AIX processes on the host by prefixing them with the shell call **/bin/sh -c**. Thus, an **on** command such as:

```
on - cal 1989
```

is executed on the host as:

```
/bin/sh -c cal 1989
```

The **ONPREFIX** variable can be used to insert a command or script between the **/bin/sh -c** call and the AIX command to be executed. For example, if **ONPREFIX** is set, the command:

```
on - cal 1989
```

is sent to the host in this form:

```
/bin/sh -c prefix cal 1989
```

where *prefix* is the value of the **ONPREFIX** variable.

This feature can be useful in the following cases:

- If you want all AIX commands initiated by **on** to execute on the host under an alternate shell such as the C shell, you can set **ONPREFIX** as follows:

```
set ONPREFIX=/bin/csh -c
```

If you want only one AIX command to run under the alternate shell, you should include the shell call in the **on** command, rather than in **ONPREFIX**, as follows:

```
on - "/bin/csh -c command"
```

where *command* is an AIX process that should be executed under the C shell.

- You can use **ONPREFIX** to specify a shell script that sets a specific AIX environment in which to run a command. This shell script should contain the line $* so that parameters are passed on to the executing shell.

  This method of setting up the AIX environment is an alternative to setting DOS environment variables and exporting them to the AIX environment.

  For example, you can set up a shell script that initializes any number of environment variables, and then performs accounting tasks. If this script is named **setup** in your personal **bin** subdirectory, set **ONPREFIX** as follows:

  ```
  set onprefix=/u/$HOME/bin/setup
  ```

  Thereafter, the command:

  ```
  on - cal 1989
  ```

  is passed to the AIX host as follows:

  ```
  /bin/sh -c /u/$HOME/bin/setup cal 1989
  ```

Setting the Terminal Emulation Environment Variable

When you leave a terminal emulation session without closing the session, the Access program creates a file called **em.ses** that contains screens of your session. The Access program uses this file to re-create your session screens when you reenter terminal emulation.

Normally, the Access program creates **em.ses** on the drive and in the directory from which you invoked the **em** command. If you change directories between one invocation of **em** and the next, the Access program either finds a different **em.ses** file and displays screens from a session other than the one you just left, or finds no **em.ses** file at all and displays a blank screen.

To avoid confusion, you can tell the Access program that you always want the **em.ses** file created and searched for in a particular location by including the following line in your **autoexec.bat** file:

```
set emses=path
```

where *path* specifies the drive, directory, and file name.

Using Terminal Emulation to an AIX/RT Host

When you use the Access program terminal emulation to log in to an AIX RT host only, you can remove the line with **nty** from your **autoexec.bat** file to save memory.

Chapter 6. The AIX Access for DOS Users vi Editor

CONTENTS

About This Chapter 6-3
Invoking the AIX Access for DOS Users vi Editor 6-4
Keyboard Features 6-4
Working Files 6-5
Configuration Options 6-5
File Recovery 6-6
Summary of Differences 6-6

About This Chapter

The Access program provides a version of the AIX **vi** editor that runs under DOS during a host file services session. The Access program **vi** editor lets you edit files on either a virtual disk or a local disk. In addition, you can specify where you want the Access program **vi** editor to keep your temporary working copy of the file—on a virtual disk or on a local disk.

Using the Access program **vi** editor generally gives you quicker response time than the AIX **vi** editor and reduces host overhead. Moreover, you do not have to learn new editor commands; the Access program **vi** editor provides most of the options and commands of the AIX **vi** editor. Those features of AIX **vi** not supported by the Access program **vi** editor are listed at the end of this chapter.

Invoking the AIX Access for DOS Users vi Editor

Before you use the Access program **vi** editor you should set the search path to include the directory where you installed **vi.exe**. It may be stored on a virtual drive or on a local drive.

Invoke the Access program **vi** editor as follows:

```
vi filename
```

where *filename* is the name of the file you want to edit. The file name can include drive and directory path specifiers.

When specifying path names with editor commands, you can use either the AIX-style directory separator, a slash (/), or the DOS-style directory separator, a backslash (\). If you use DOS-style directory separator within the **vi** editor, you must use two backslashes (\\). You cannot specify wildcard characters (* or ?) or metacharacters (for example, `[a-z]`) as part of the file name.

If the file you are editing is larger than 16,000 lines, you should invoke the Access program **vi** editor in the **ex** mode as follows:

```
vi -e filename
```

If you want to edit a file in DOS format and have it saved in DOS format, invoke the Access program **vi** editor as follows:

```
vi -d filename
```

If you invoked the editor in DOS mode and want to switch to AIX mode, or if you invoked the editor in AIX mode and want to switch to DOS mode, you can change the mode by setting a new **vi** configuration option, called **dosmode**. For example, to set **dosmode** while in **vi**, enter the **:set** subcommand as follows:

```
:set dosmode
```

which can be abbreviated to `:set dm`.

If you follow this with the **:e!** subcommand, you can reedit the file, reading it in as DOS mode. However, this will discard any changes already made. Use **:set nodosmode** or **:set nodm** to change to AIX mode while in **vi**.

By setting **dosmode** in the file **ex.rc**, you can have the Access program **vi** editor check the mode whenever a file is read or written. For more information on setting **dosmode** and other options, see "Configuration Options" on page 6-5.

Keyboard Features

The following keys on your personal computer keyboard have special meanings for the Access program **vi** editor:

- The **Page Up (PgUp)** key generates a **vi Ctrl-B** command (scroll back one screen).
- The **Page Down (PgDn)** key generates a **vi Ctrl-F** command (scroll ahead one screen).
- The **Insert (Ins)** key puts **vi** into the input state.
- The **Delete (Del)** key generates a **vi x** command (delete the current character).

In addition, pressing a function key is the equivalent of typing a pound sign (#) followed by the number of the corresponding function key. For example, pressing **F6** or typing `#6` invokes the same predefined macro for any function defined for the **F6** key.

Working Files

Like the AIX **vi** editor, the Access program **vi** editor makes a temporary working copy of your file. You can choose the directory that you want to contain these working files, which can be on either a virtual drive or a local drive. (The Access program **vi** editor names temporary files with a **vi** or **ri** prefix, instead of the AIX **vi** prefixes **Ex** and **Rx**.)

The Access program **vi** editor usually creates the working files in the **/tmp** directory on the virtual drive if you have a host file services session in progress. If not, the working files are created in the root directory of the current drive.

Note: On a multi-host system, make sure that you always end your Access program **vi** session before logging off the virtual drive. If you log off the virtual drive before ending your Access program **vi** session, temporary files on the virtual drive are not closed properly and the Access program stops working.

Although there can be more room for the working files on a virtual drive, you can generally improve the performance of the Access program **vi** editor by forcing it to create the working files on a local drive. Performance always improves when your local drive is a fixed disk. When the local drive is a diskette drive, the improvement depends on the AIX system load and the amount of memory in your personal computer.

You can specify a particular directory (either on a virtual drive or a local drive) for working files by setting the **vi** configuration option **dir** to that directory. The section "Configuration Options" describes this and other options.

Configuration Options

You can specify most of the configuration options for the Access program **vi** editor that are available for the AIX **vi** editor. If you specify the options in the file **ex.rc**, the Access program automatically sets them when you invoke the **vi** editor.

Add the following line to **autoexec.bat** to specify the directory where the configuration file **ex.rc** is stored:

```
set HOME=directory
```

You must fully qualify this directory name, including the drive specification.

Create the file called **ex.rc** in this directory and specify any **vi** configuration options you desire. For example, to specify where you want **vi** to create working files, add the following line to **ex.rc**:

```
set dir=directory
```

where *directory* is a fully qualified directory name, including the drive letter (either a local or virtual drive). Since backward slashes (\) are used as quote characters in configuration options, the slash (/) is the preferred directory separator. If you use backward slashes, you must double them. (Remember, if you specify a virtual drive, you must have a host-file services session in progress when you invoke the Access program **vi** editor.)

The Access program provides the new **dosmode** option for **vi**. When **dosmode** is set, the Access program **vi** editor assumes that text files are in DOS format and translates carriage-return and line-feed pairs into single line feeds when they are read. When writing files in DOS mode, the Access program **vi** editor translates line feeds into carriage-return and line-feed pairs.

To set DOS mode automatically when you invoke **vi**, enter the following line in **ex.rc**:

```
set dosmode
```

To ensure AIX-mode editing, enter:

```
set nodosmode
```

The following AIX **vi** configuration options are not supported by the Access program **vi** editor:

- The **hardtabs** option (there is no need to change these values in the Access program **vi** editor).
- The **timeout** option (macros always behave as if **notimeout** was set).

File Recovery

The Access program **vi** editor provides an expanded capability for file recovery, since the standard AIX **vi** editor recovery facilities cannot handle recovery when your session is interrupted by one of the following:

- Host connection failure
- Personal computer problem
- Creating temporary files on a local drive.

Use the following procedure to recover files:

1. Reestablish your host-file services session.
2. Determine which files the Access program can recover by using the **exrecovr** command. The **-r** flag causes **exrecovr** to list all of the text files you were editing that can be recovered. Enter the **exrecovr** command in the following manner:

   `exrecovr -r [`*`directory...`*`]`

   where *directory* specifies the names of the directories that **exrecovr** should search for temporary working files. If you omit the *directory* parameter, **exrecovr** searches the files **\tmp** and **\usr\preserve** on the virtual drive and the root directory of all local drives. (**\usr\preserve** contains files that the AIX host would normally have deleted from **\tmp** during host recovery.)

   If you specified another directory for working files, you should specify the name of that directory when you issue the **exrecovr** command.
3. Using the list of files provided in step 2, recover each file by entering the following at the command line:

   `exrecovr` *`directory filename > recoverfile`*

   The *directory* parameter is the fully qualified directory name that contains the recoverable file, and the *filename* parameter specifies the text file that was being edited. Redirect the recovered text to a file you name with the *recoverfile* parameter. This parameter can be a fully qualified file name.
4. Compare your original text file to the recovered version to determine which editing changes have been applied to the text file and what additional changes you still need to make.

Summary of Differences

The differences between the AIX **vi** editor and the Access program **vi** editor include the following:

- The Access program **vi** editor does not support the AIX **vi** filters (for example, `:1,10! sort` and `:w!`).
- The Access program **vi** editor names working files with **vi** and **ri** as prefixes. AIX **vi** uses **Ex** and **Rx** as prefixes.
- The erase (**Ctrl-H**) and kill (**Ctrl-X**) functions are fixed in the Access program **vi** editor. They cannot be redefined as in the AIX **vi** editor.

- The AIX **vi** configuration options **hardtabs** and **timeout** are not supported by the Access program **vi** editor.

Since backward slashes (\) are used as quote characters by **vi**, the slash (/) is the preferred directory separator. If you use backward slashes, you must double them.

Chapter 7. Command Summary

CONTENTS

About This Chapter 7-3
Syntax Diagrams 7-4
Command, Flag, and Parameter Notation 7-6
aix2dos 7-7
doswhat 7-9
dos2aix 7-10
em 7-11
exrecovr 7-12
jobs 7-13
kill 7-14
login 7-16
logout 7-18
nty 7-19
on 7-20
- on Command Formats 7-20
- Search Path and Environment Considerations 7-21
- Pipes and Redirection 7-22
- Standard Input, Standard Output, and Standard Error 7-22
- Breaking Out of on Jobs 7-22
- Detached Jobs 7-23
- Job Control 7-23

pciinit 7-24
printer 7-25
setdebug Command 7-28
udir 7-32
vi 7-33

About This Chapter

This chapter presents a detailed description of the Access program commands. The commands are presented in alphabetical order, and each description includes the purpose, format, and remarks about the command.

Syntax Diagrams

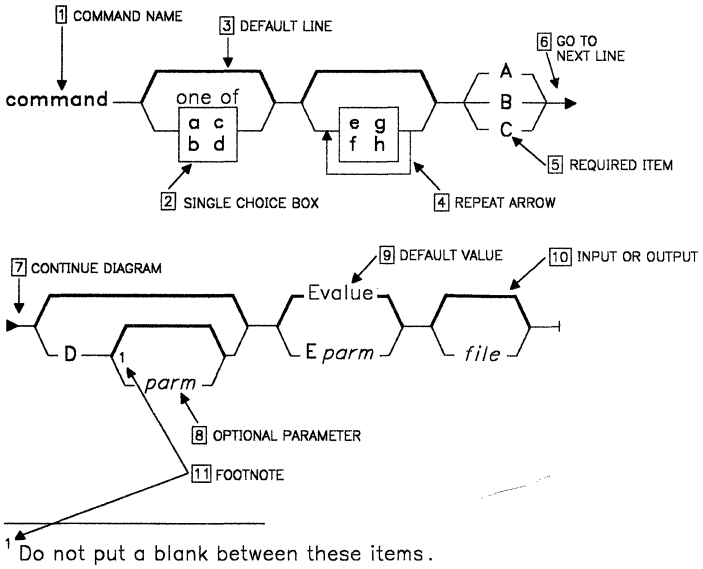
Before each command discussion in this chapter is a syntax diagram. These diagrams are designed to provide information about how to enter the command on the command line. A syntax diagram can tell you:

- Which flags can be entered on the command line
- Which flags must take parameters
- Which flags have optional parameters
- Default values of flags and parameters, if any
- Which flags can and cannot be entered together
- Where you must enter flags or parameters and where you have a choice
- Where you can repeat flag and parameter sequences.

The syntax diagrams conform to the following conventions:

- Diagram items that must be entered literally on the command line are in **bold**. These items include the command name, flags, and literal characters.
- Variable diagram items that must be replaced by a name are in *italics*. These items include parameters that follow flags and parameters that the command reads, such as *files* and *directories*.
- Default values that do not have to be entered are in normal font on a **bold** path.

The following diagram is an example that illustrates the conventions used in the syntax diagrams. Each part of the diagram is labeled. An explanation of the labels follows the diagram.



OL805370

You interpret the diagram as follows:

1 COMMAND NAME The first item in the diagram is the name of the command you want to invoke. It is in bold, so it must be entered exactly as it is shown in the diagram.

After the command name, the path branches into two paths. You can follow either path.

2 SINGLE CHOICE BOX

If you follow the lower path, you encounter a box with the words **one of** over it. You can choose only one item from this box.

3 DEFAULT LINE If you follow the upper path, you bypass the single choice box, and enter nothing. The bold line around the box is a default line, which means that you do not have to enter anything from

that part of the diagram. Exceptions are usually explained under "Description." One important exception, the blank default line around input and output files, is explained in item **10**.

4 REPEAT ARROW When you follow a path that takes you to a box with an arrow around it, you must choose at least one item from the box. Then you can either follow the arrow back around and continue to choose items from it, or you can continue along the path. When following the arrow around just the box (rather than an arrow that includes several branches in the diagram), do not choose the same item more than once.

5 REQUIRED ITEM Following the branch with the repeat arrow is a branch with three choices and no default line around them. This means that you must choose one of **A**, **B**, or **C**.

6 GO TO NEXT LINE If a diagram is too long to fit on one line, this character tells you to go to the next line of the diagram to continue entering your command line. Remember, the diagram does not end until you reach the vertical mark.

7 CONTINUE DIAGRAM

This character shows you where to continue with the diagram after it breaks on the previous line.

8 OPTIONAL PARAMETER

If a flag can but does not have to take a parameter, the path branches after the flag. If you cannot enter a space between the flag and parameter, you are told in a footnote.

9 DEFAULT VALUE Often, a command has default values or actions that it will follow if you do not enter a specific item. These default values are indicated in normal font in the default line if they are equivalent to something you could enter on the command line (for example, a flag with a value). If the default is not something you can enter on the command line, it is not indicated in the diagram. However, it is discussed under "Flags."

Note: Default values are included in the diagram for your information. Do not enter them on the command line.

10 INPUT OR OUTPUT A command that can read either input files or standard input has an empty default line above the file parameter. If the command can write its output to either an output file or to standard output, it is also shown with an empty default line above the output file parameter. If a command can read only from standard input, an input file is not shown in the diagram, and standard input is assumed. If a command writes only to standard output, an output file is not shown in the diagram, and standard output is assumed. When you must supply a file name for input or output, the file parameter is included in the diagram without an empty default line above it.

11 FOOTNOTE If a command has special requirements or restrictions, a footnote calls attention to these differences.

Following are examples of valid ways this command can be entered based on this syntax diagram.

```
command name A
command name C
command name a B
command name d B
command name e A
command name e g f A
command name C D
command name C D8
command name A E7
command name B myfile
command name a e g B D3 E6 myfile
command name d f e h C D myfile
```

When the order of flags is important, it is indicated in the diagram, under "Flags," or in both places. Otherwise, the flags can be entered in any order. With this in mind, an additional example of how to enter this command is:

```
command name E9 a D g A h f myfile
```

Command, Flag, and Parameter Notation

The following type style conventions are used in command descriptions to distinguish different kinds of information:

bold Commands, flags, and other items in bold are to be entered literally.

italics Command parameters, flag parameters, and other items in italics are items for which you substitute an appropriate value in that position on the command line. For example, if you see *file*, you should type in the name of a file in that position.

[ ] Items in brackets are optional. The only exception is brackets that are in bold. Brackets in bold are part of what should be entered literally.

. . . Items followed by an ellipsis can be repeated. Thus, if you see *file* . . . , you can type several file names separated by blanks.

Using these conventions, the following string:

-D*name*[=*value*]

shows that, with the **-D** flag, the *name* parameter is required but assigning a *value* to *name* is optional. The following are valid ways to specify this flag and parameter combination:

```
-Daxis
-Daxis=10
```

The next string shows a parameter that can be replaced by several values:

-l *filep* . . .

The following are valid ways to enter the **-l** flag:

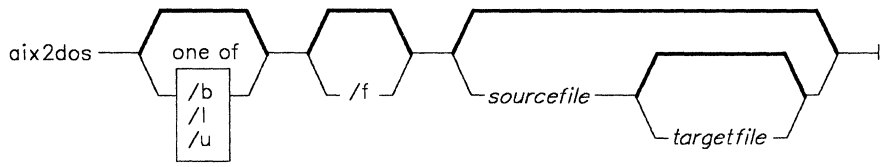
```
-l memo letter
-l memo
-l letter
```

aix2dos

Purpose

Converts a text file from AIX format to DOS format.

Format



A5ACB007

Description

The **aix2dos** command converts AIX format files whose lines end in a new-line sequence to DOS format files that use a carriage return to indicate the end of the line. When used without parameters, **aix2dos** reads from standard input and writes to standard output. You can also specify *sourcefile* and *targetfile*. If only *sourcefile* is specified **aix2dos** writes the result to standard output.

Notes:

1. The *sourcefile* and *targetfile* must be different files.
2. If the source file is already in DOS style, **aix2dos** does not alter the format of the file.

Normally, **aix2dos** removes all extra carriage-return characters preceding the end-of-line sequence before doing the end-of-line conversion. The **/f** flag prevents the removal of these extra carriage returns and forces the translation of each line-feed character into a carriage-return character. Although not normally used, this flag preserves unusual end-of-line sequences to ensure that the file can be translated from AIX to DOS and back without any changes.

The **aix2dos** command can be used in combination with DOS and AIX pipes and redirection. For example, the command:

```
aix2dos textfile | sort > newtext
```

converts the AIX file **textfile**, sorts the converted text, and writes it to the DOS file **newtext**.

Flags

/b Preserves 8-bit (binary) character representations. When this flag is not specified, **aix2dos** truncates all characters to 7-bit ASCII representation.

Note: If the *sourcefile* contains binary characters and **/b** is not specified, **aix2dos** will notify the user.

/f Prevents the removal of extra carriage returns in front of the end-of-line sequence, and forces the translation of each line-feed character into a carriage-return character.

/u Converts text to uppercase.

/l Converts text to lowercase.

Notes:

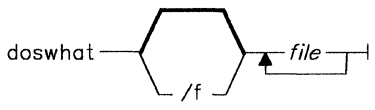
1. The **/b**, **/u**, and **/l** flags are mutually exclusive.
2. When neither **/u** nor **/l** is specified, the case of each text character is unchanged.
3. When issued from the AIX Operating System, the **/b**, **/u**, **/l**, and **/f** flags must be specified with the AIX hyphen (-) rather than the DOS slash (/). You can use either the slash or the hyphen from the DOS operating system.

doswhat

Purpose

Displays identifying information in DOS or AIX files on the local or virtual drive.

Format



A5ACB008

Description

The **doswhat** command behaves like the AIX **what** command except that it executes under DOS and operates on DOS or AIX files on either a virtual drive or a local drive. The **doswhat** command searches the specified files for the character sequence `@(#)` and prints the text following the character sequence up to the first double quote ("), greater than (>), new line, backslash (\), or null character.

You can invoke **doswhat** on source or executable files. File names can contain DOS wild card characters.

The **doswhat** can check the version number or other identifying information of executable DOS files. Executable files are those with the extensions **.exe**, **.com**, **.bat**, or **.sys**. For example, to find out identifying information about **bridge.sys**, enter the following at the command line:

```
doswhat bridge.sys
```

Information concerning the release number, date of creation, and other useful information is listed on the display.

Flags

/f Causes **doswhat** to report if *file* is a DOS or AIX ASCII file, binary file, directory or AIX device.

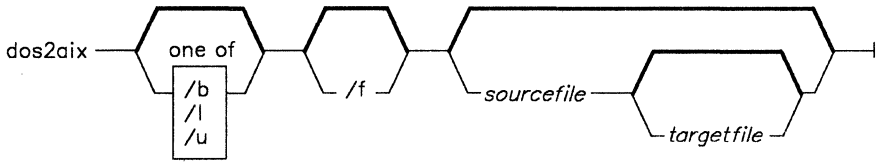
For more information see the **get** command in *AIX Operating System Commands Reference.*

dos2aix

Purpose

Converts a text file from DOS format to AIX format.

Format



A5ACB009

Description

The **dos2aix** commmand converts a file from DOS format that uses lines ending in a carriage-return and line-feed sequence, and **Ctrl-Z** as an end-of-file indicator to AIX format that uses lines ending in a new-line sequence, and and ASCII line-feed. When invoked without parameters, **dos2aix** reads from standard input and writes to standard output.

When you specify *sourcefile*, **dos2aix** reads from the specified file and writes to standard output. To write the output of **dos2aix** to a file you must specify both a *sourcefile* and a *targetfile*. It is important to remember that the source and target file must have different names.

Note: If a source file is already in AIX style, **dos2aix** does not alter the format of the file.

The **dos2aix** command normally removes all extra carriage-return characters preceding the end-of-line sequence before doing the end-of-line conversion. The **/f** flag prevents the removal of these extra carriage returns and forces the translation of only the final carriage-return and line-feed sequence into a line feed. This flag preserves unusual end-of-line sequences to ensure that the file can be translated from AIX to DOS and back without any changes.

You can use the **dos2aix** command in combination with DOS and AIX pipes and redirection. For example, the command:

```
sort < text | dos2aix > newtext
```

sorts the DOS file **text**, converts the sorted text, and writes it to the AIX file **newtext**. It is important to note that you cannot redirect a converted file to itself.

Flags

/b Preserves 8-bit (binary) character representations. When this flag is not specified, **dos2aix** truncates all characters to 7-bit ASCII representation.

/f Prevents the removal of extra carriage returns in front the end-of-line sequence, and forces the translation of only the final carriage-return and line-feed sequence into a line feed.

/u Converts text to uppercase.

/l Converts text to lowercase.

Notes:

1. The **/b**, **/u**, and **/l** flags are mutually exclusive.
2. When neither **/u** or **/l** is specified, the case of each text character is unchanged.
3. When issued from the AIX Operating System, the **/b**, **/u**, /l, and **/f** flags must be specified with the AIX hyphen (-) rather than the DOS slash (/). You can use either the slash or the hyphen from the DOS operating system.

em

Purpose

Invokes the Access program terminal emulator.

Format

A5ACB003

Description

The **em** command presents you with a choice of AIX host connection paths over which to establish a terminal emulation session. When you establish a terminal emulation session over an RS-232 connection path, **em** also allows you to alter certain default connection parameters to match the host settings determined by the person who administers your system.

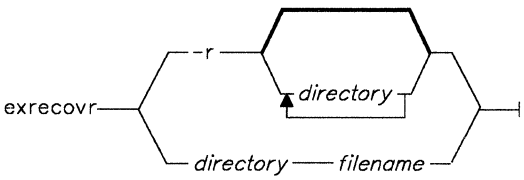
Once your terminal emulation connection has been established, log in to your AIX host and conduct your AIX session, as you would over a standard VT100 terminal, or a DOS session, as you would from a PC console. For more information, see Chapter 4, "Using Your Computer as an AIX Terminal."

exrecovr

Purpose

Provides a method for the **vi** editor to recover files after a system failure.

Format



A5ACB002

Description

The **exrecovr** command is used to recover files that were being edited by the Access program **vi** editor when a system failure occurred.

The **exrecovr -r** command, when invoked without additional parameters, searches the **\tmp** and **\usr\preserve** directories on the virtual drive (if a host file services session is in progress), and the **root** directory of all local drives.

Note: If you set a default **\tmp** directory with the **HOME** environment variable, you must explicitly specify the full path name of this directory when using the **exrecovr** command.

To recover a specific file, specify *filename* as displayed by an **exrecovr -r** command and use redirection symbols to store the recovered file in a new file. The new file name may be fully qualified.

For example, to recover a file called **myfile** in the **\usr\preserve** directory you would enter the following:

```
exrecovr \usr\preserve myfile > rcvrfile
```

In this example, **myfile** was the file being edited when the system failed. The file **rcvrfile** is used to capture the output recovered by **exrecovr**.

Note: When you recover files with **exrecovr**, you must compare your original file to the recovered file to determine which editing changes were made before the system failure.

Flags

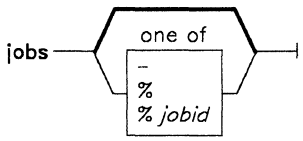
-r Lists the files that Access program program **vi** editor can recover. When you specify a *directory*, **exrecovr** only lists the recoverable files in that directory. You must use a dash (-) as a switch for this flag.

jobs

Purpose

Displays the job table of **on**-initiated AIX tasks, clears it of completed tasks, and reattaches the DOS console to detached jobs.

Format



A5ACB004

Description

When invoked without parameters, the **jobs** command displays the current job table. The job table is displayed in the following form:

```
JOB     HOST        STATE       EXIT STATUS   COMMAND
[1]     host1       RUNNING                   AIXcommand1
[2]     host2       DONE        exit (0)      AIXcommand2
```

Entering `jobs -` clears completed entries from the job table and removes any temporary files on the host associated with each task. When a job is cleared, **jobs** discards any standard output that was saved for viewing after reattachment.

You can reattach the DOS console to the specified detached job by entering `jobs %`*jobid*. The *jobid* parameter is the job number from the job table. If you enter the command sequence without a *jobid* number after the percent sign (%) you will be reattached to the lowest numbered task in the job table.

When you reattach to a running task, the non-redirected standard output of the AIX process is displayed on your screen. When you reattach to a complete task, the entire non-redirected standard output from the beginning of the process is displayed. You can reattach any number of times to a task while it is running and still see the entire output from the start of the task when you reattach after a task completes.

Detached jobs can be halted and cleared from the job table by reattaching, pressing **Ctrl-C** or **Ctrl-Break**, then responding to the resulting prompt. To halt the AIX process, use **kill**.

The **jobs** command is built into the **on** command. This means that if you want to execute an AIX command called **jobs**, you must use the form **on - jobs** rather than linking **jobs.exe** to **on.exe**.

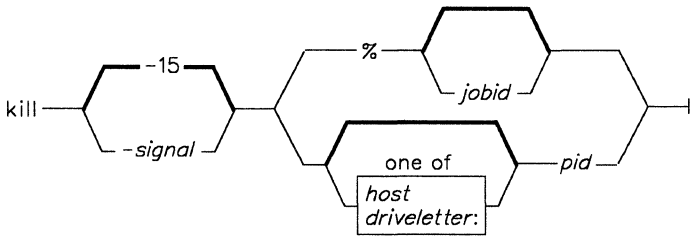
You can use the **ONNAME** environment variable to change the name of the **jobs** command. See "The .exe Form of the on Command" on page 7-21 for more about the **ONNAME** environment variable.

kill

Purpose

Sends signals to AIX processes, including those initiated by **on**.

Format



A5ACB005

Description

The Access program **kill** command is similar to the AIX **kill** command. If you specify a *jobid* with the Access program **kill** command, the job table entry for the specified job is cleared. Specifiying the *jobid* kills not only the original process, but also any child processes the job may have generated.

You can also use **kill** and specify the AIX process ID (*pid*) displayed after issuing an **on** command.

Note: If the **on**-initiated detached job generated child processes and you used the **kill** *pid* form of the **kill** command, it does not kill the entire process group. You must issue **on - ps** to determine the process IDs of the child processes and kill each child process explicitly.

If no *jobid* is specified after the percent sign (%), the signal is sent to the lowest-numbered job in the job table, but the job table entry is unchanged.

You can use the **ONNAME** environment variable to change the name of the **kill** command. See "The .exe Form of the on Command" on page 7-21 for more about the **ONNAME** environment variable.

Note: The Access program **kill** command uses the **on** command. Do not use the AIX **kill** command to stop background processes running on a virtual DOS drive.

Flags

-signal Specifies the AIX signal to be sent to the AIX process. The default signal is 15. If the default fails to kill the process, a signal of 9 can be used. A dash (-) must be used as a switch for this flag.

jobid Specifies the job control ID number of an **on**-initiated detached job as reported in the job table.

host Specifies the name of the AIX host where the signal is to be sent. If neither *host* nor *driveletter* is specified, the host associated with the current drive is used.

driveletter: Specifies the drive letter of the virtual drive associated with the AIX host where the signal is to be sent. If neither *host* nor *driveletter* is specified, the host associated with the current drive is used.

- Indicates that the signal is to be sent to the AIX host associated with the current drive. This parameter is not entered on the command line. If neither *host* nor *driveletter* is specified, the host associated with the current drive is used.

pid Specifies an AIX task's process ID number, that can be obtained with the AIX **ps** (process status) command. If *pid* is negative or if *%jobid* is specified, the entire process group is killed.

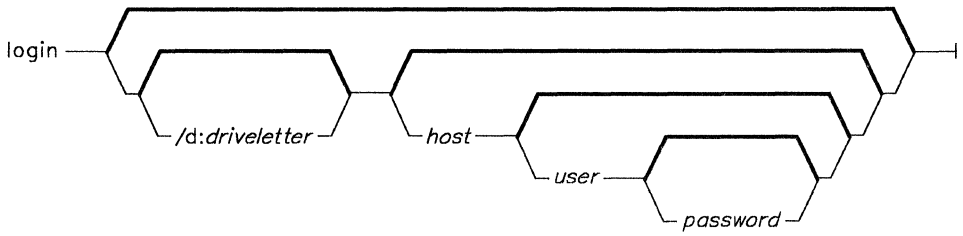
For additional information see **kill** in the *AIX Operating System Commands Reference.*

login

Purpose

Initiates the Access program host file services session.

Format



A5ACB006

Description

The **login** command initiates a file services session by making a host available as a virtual drive. When you issue the **login** command, you are presented with a choice of AIX hosts and connection paths over which to establish a host file services session. After you choose a host connection path, the system prompts you for your AIX user name and password. If you select RS-232 (com*n*) as your connection path, the system prompts you to select the correct baud rate for your machine. After you enter the correct baud rate, a terminal emulation window appears and you can enter your user name and password.

Once the Access program has established the host connection, **login** indicates the virtual drive on which the AIX file system is available during the host file services session. For example:

```
AIX File System Available as Drive D:
```

By default, the Access program makes the selected host available to you as the next available drive under the following conditions:

- Drive C:, if your personal computer has only two diskette drives
- Drive D:, if your personal computer has a fixed drive
- The next driveletter available as determined by those already in use.

You have the option of specifying the drive you want associated with a particular host by specifying the drive as part of the **login** command, as in the following example:

```
login /d:e
```

Note: If you try to log in to a given host through more than one drive, the Access program displays an error message.

When you log in over a LAN connection path, you can specify the host you want as part of the **login** command, bypassing the Host-selection menu. You can include your user name or your user name and password as part of the command line. The Access program prompts you for your user name and password if you omit them.

When you log in over an RS-232 connection path, do not specify a login name or password from the command line. Instead, select an RS-232 connection path from the command line using the following form:

```
login comn
```

or

```
login /d:driveletter comn
```

The Access program prompts you for the baud rate and places you in a terminal emulation window to enter your user name and password. At this point **login** may be canceled by

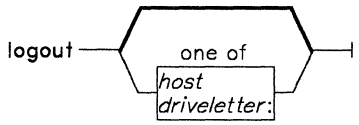
pressing **F10**. Once the login is complete, the file services session must be started by pressing **F9**.

logout

Purpose

Terminates one or more of the Access program active host file services sessions.

Format



A5ACB012

Description

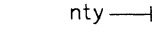
To log out of all currently established host file service sessions, enter the **logout** command with no parameters. Specify the *host* parameter to log out of a file services session on a particular host. You can also specify *driveletter*: to log out of a file services session on a particular host.

nty

Purpose

Initializes the Access program terminal emulation.

Format



A5ACB013

Description

The **nty** command (along with the **pciinit** command) is inserted into the **autoexec.bat** file when you install the Access program files. Once inserted into the **autoexec.bat** file, the **nty** command is executed each time you start your personal computer.

You can execute the **nty** command manually, rather than have it automatically run when you start your system. However, if you are running DOS Services on a PS/2 you must run **nty** before you attempt to start a terminal emulation session.

Note: This command is not applicable for terminal emulation with an IBM RT host computer. If you want to establish a terminal emulation session with an RT workstation only, this command is not required.

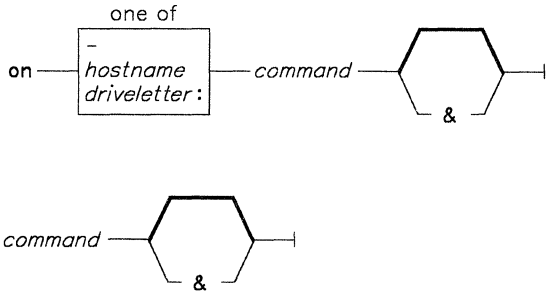
If you do not intend to use terminal emulation, you may remove the **nty** command from the **autoexec.bat** file.

on

Purpose

Executes AIX commands on a specified host computer.

Format



A5ACB014

Description

The **on** command allows you to execute non-interactive AIX commands from the DOS environment. The commands are executed under the AIX shell. The current directory of the AIX process is the same as the current directory of the selected virtual drive. Standard output produced by a command displays on the DOS screen unless it is explicitly piped or redirected elsewhere. Screen output returned to the DOS environment by **on** is automatically converted to DOS format. The Access program converts AIX-style text with lines separated by line-feed characters to DOS-style text by adding carriage-return characters between lines.

on Command Formats

There are two forms of the **on** command. One form uses **on** as a prefix to the AIX command. The second form creates a **.exe** form of the AIX command using a copy of **on.exe**.

Using on to Prefix Commands

To use the prefix form of the **on** command, enter **on** followed by *driveletter*:, *hostname,* or a dash (-). Use *driveletter*: or *hostname* to specify the drive or host on which to execute the AIX command. If you enter a dash, the AIX command is executed on the current virtual drive. The *hostname* parameter must be typed exactly as shown on the host-selection screen. For example, the following entry:

```
on zeus cal 1989
```

runs the AIX **cal** command on the host called `zeus` and displays the screen output produced by **cal** on your DOS screen.

Note: If you are connected to a host file service session over an RS-232 connection path, type `com`*n* in place of the host name, where `com`*n* is the RS-232 port.

To run the **cal** command on the virtual drive D, enter the following at the command line:

```
on d: cal 1989
```

To run the same command on the host associated with the current drive, enter the following at the command line:

```
on - cal 1989
```

You can issue any valid non-interactive AIX command with valid options using the **on** prefix form. To prevent ambiguity in interpreting the command, put an AIX-style escape character in front of any special characters.

The following command executes the AIX **date** command and places the output in the file **temp** in the current working directory on the host:

```
on host1 date "\>" temp
```

However, the command:

```
on host1 date > temp
```

places the output of the **date** command in the file **temp** in the current working directory on your personal computer.

Using **on**, you can issue any non-interactive AIX command that is not a valid DOS name. Do not use DOS mapped names with the **on** command.

The .exe Form of the on Command

The **.exe** form of the **on** command lets you omit the word **on** from the command line and only type the AIX command name. It is important to note, this form of the **on** command only runs on the virtual drive. To use the **.exe** form of **on**, first copy or link the **on** file (**on.exe**) to the name of the AIX command you want to execute from the DOS environment. For example, to create a DOS executable version of the AIX command **ls**, enter the following at the DOS prompt:

```
copy on.exe ls.exe
```

Alternately, you can use the AIX **ln** command to link the **on.exe** command file to the AIX command. First, link **on.exe** to **ln.exe** by entering the following:

```
on - ln on.exe ln.exe
```

Then, use the **ln.exe** command to link files in the DOS environment to those in the AIX environment.

Note: You cannot link copies across AIX filesystem boundaries.

Once **on.exe** is copied or linked and renamed to resemble the AIX command, you can run the AIX command directly from the DOS prompt. For example:

```
copy on.exe cal.exe

cal 1989
```

When you use the **on.exe** form of the, the AIX command you create must be in lowercase (the convention for commands in AIX), and must conform in other respects to the DOS rules for file names.

To initiate multiple AIX commands, enter each **on** command individually. They cannot be entered as one command sequence with the commands separated by semicolons.

Search Path and Environment Considerations

The **UPATH** environment variable must be able to find the AIX command created with the **on** command.

The AIX search path used by **on** is specified by the **PATH** environment variable. To change this path, set the DOS environment variable **UPATH**. Use the same syntax for **UPATH** that you use with the AIX **PATH** environment variable. The syntax for environment variables includes colons and forward slashes. For example, to set a search path to the directory called `/u/joe/bin`, enter the following at the command line:

```
set upath=:/bin:/usr/bin:/u/joe/bin
```

The value of the DOS **UPATH** variable is assigned to the AIX **PATH** environment variable when **on** is run.

To set any other AIX environment variables required by AIX commands invoked with **on**, use the DOS **set** command to set the variables in the DOS environment. For example:

```
set home=/usr/paula
set tz=PST8PDT
set lpdest=laser2
```

After setting the DOS environment variables, set the DOS **EXPORT** environment variable equal to the list of environment variables to be communicated to AIX.

```
set export=home tz lpdest
```

Since DOS environment variable names are always stored as uppercase, **on** converts all environment variable names to uppercase when they are placed in the AIX environment. Thus, only all-uppercase variable names can be exported to the AIX environment. However, the values of the DOS variables are exported literally.

Pipes and Redirection

You can use pipes and redirection with AIX or DOS. The following is a list of the pipes and redirection characters you can use.

Character	Description
< and >	Standard characters which redirect DOS input and output.
|	Pipes DOS output.
{ and }	Redirects AIX input or output.
! or ^	Pipes AIX output.

Standard Input, Standard Output, and Standard Error

The Access program transfers input to and output from the AIX command between AIX and DOS using temporary files created in the directory **/tmp** on the host.

If standard input on DOS is redirected (using <), the standard input is copied to a temporary file on the host. Standard input to the AIX command is then redirected from this temporary file. You can use the DOS **<CON** redirection argument to redirect input from the keyboard. Since the standard input file is created and closed before the AIX command is run, you cannot enter input to the AIX command interactively.

In the absence of any output redirection on AIX, both standard output and standard error from the AIX command are redirected to a file named *pid*.chm, where *pid* is the process ID number of the AIX command. The contents of this file are copied to standard output on DOS as it is created on AIX.

Upon completion, **on** deletes all temporary files created on the host. If you have detached an **on**-initiated AIX command, the temporary standard output file is saved until you reattach to the task or clear it from the Job table. The output is displayed on your screen.

Breaking Out of on Jobs

Pressing **Ctrl-Break** or **Ctrl-C** while **on** is running causes the following prompt to be displayed:

```
a - abort, c - continue, d - detach:
```

Pressing a stops **on** in the DOS environment and removes the task from the Access program job table. Pressing c causes **on** to continue from the point of interruption. Pressing d detaches the task.

Detached Jobs

You can run an AIX program as a detached job by terminating the command line with an ampersand (&), or by responding with a d to the `abort, continue, detach` prompt described above.

When you detach a task, **on** reports both an **on** job ID number and the process ID number of the AIX command. **on** then exits immediately back to DOS.

When **on** is used in this way, temporary files are not deleted automatically. If you have not redirected the standard output of the AIX command, it is saved for review at the time of reattachment.

If you detach **on** commands containing DOS pipes or redirection, the results might not be what you expect. For more information refer to Chapter 3, "The on Utilities."

Job Control

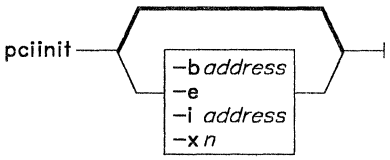
The **on** and **jobs** command, give you control over detached jobs similar to that offered by the AIX C shell. See Chapter 3, "The on Utilities" and the command summary entries for **jobs** and **kill** for more information.

pciinit

Purpose

Initializes the Access program software.

Format



A5ACB010

Description

When you install the Access program using the **install** program, **pciinit** is included in the **autoexec.bat** file on your Access program working diskette so that initialization occurs automatically when you start your personal computer.

Note: If you have a LAN card the install script will prompt you for the proper interrupt level at the time of installation. To find out your interrupt level, ask the person who administers your system.

Flags

-b Indicates the network address the Access program uses to send a broadcast request for a host-selection map. This flag is optional, and you should not use it unless the person who administers your system instructs you to.

-e Returns the current network address setting.

-i Indicates the internet address of your personal computer in the following form:

xxx.xxx.xxx.xxx

where *xxx* is a number between 0 and 255.

When you install the Access program with the **install** program, **install** prompts you for the internet address of your system. If you do not know your internet address, ask the person who administers your system.

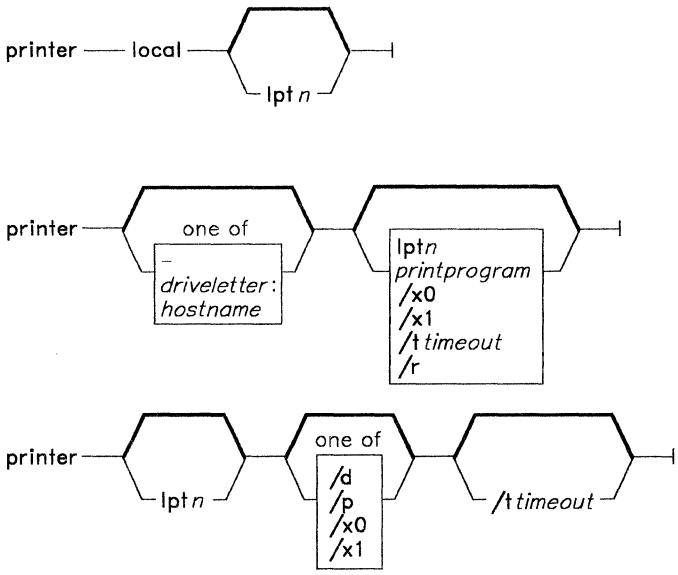
-x*n* Indicates which interrupt channel the network interface adapter uses, where *n* equals the channel number plus an offset of 8. Omitting this flag is equivalent to specifying **-x10** (the default interrupt channel 2).

printer

Purpose

Directs printing to a local printer attached to your personal computer or to a remote AIX printer.

Format



A5ACB011

Description

The **printer** command allows you to send output to local printers or printers on the host system. The **local** parameter specifies that print requests are sent to a local printer attached to your personal computer. Local print requests can be issued in any one of the following ways:

- Using the DOS **print** command.
- Using the **Print Screen** key (**Shift-PrtSc** for PCs, XTs, and ATs).
- Using the DOS **copy** command and specifying the local printer device (PRN) as the target.
- Invoking the printer from an application program.

To use a printer on a remote AIX host, use the second form of the **printer** command. Use *driveletter*: or *hostname* to specify the remote host. For example:

```
printer e: lpt2
```

sends all print requests directed to LPT2 to the printer associated with the host attached to drive E, while:

```
printer frodo lpt2
```

sends all LPT2 print requests to the printer associated with the host named `frodo`.[1]

If you type a hyphen (-) rather the *driveletter*: or *hostname* parameters, your printing is directed to the current host.

---

1 You cannot use a host named `local`.

Note: If you select the current host with the **printer** command, then change the current host by changing your working drive, printing remains directed to the original default host until you enter a new **printer** command. For example,

```
printer - lpt2
```

sends all LPT2 print requests to the printer on the host associated with current drive.

Your print requests are sent to one of three print streams (LPT1, LPT2, or LPT3) and spooled for printing on the printer attached to the specified host. When you omit the print stream identifier from the **printer** command, LPT1 is assumed.

Remote print requests sent to AIX printers can be issued in any one of the following ways:

- Using **Print Screen** (**Shift-PrtSc** for PCs, XTs, and ATs) to print the current screen contents. When you use this method, printing is always directed to the LPT1 print stream.
- Pressing **Ctrl-Print Screen** (**Ctrl-PrtSc** for PCs, ATs, and XTs) to print continuous screen contents. If you use this print method, you must first use the **printer /x0** command.
- Using the DOS **copy** command to copy a file to one of the three print streams LPT1, LPT2, or LPT3.
- Invoking the print function from within a DOS application.

Note: The DOS **print** command is not recommended with remote printing because it uses a spooler that could interfere with the Access program. Use the **copy** command instead.

The **printer** command allows you to specify an AIX command or program for remote printing. The *printprogram* parameter defines an AIX command string to be associated with the specified print stream. For example:

```
printer - lpt3 print
```

causes all subsequent print requests directed to LPT3 to be sent to the **print** program on the current default host and printed on the printer associated with the **print** command.

With the *printprogram* parameter, you can specify any AIX command, not only print commands. With this parameter, you can include command options as well as pipes and redirection characters. When the command string contains spaces, it must be enclosed in quotes. For example:

```
printer - lpt2 "spell | /usr/pci/bin/aix2dos > sp.err"
```

causes output directed to LPT2 to be sent to the AIX spelling check program, with the errors converted to DOS format and written to a file called **sp.err**.

When you do not use the *printprogram* parameter to specify a print command for a particular print stream during a host file services session, the Access program sends print requests for that print stream to the system default print command for output to the default system printer.

When you invoke remote printing, the Access program buffers print requests but does not send them to the **print** command until you close the print stream by exiting the DOS application or issuing another DOS command.

The third form of the **printer** command is used to modify the existing settings of the print streams. For example, if you have issued the following command:

```
printer - lpt3 print
```

and later want to change the timer interrupt option, you could type:

```
printer lpt3 /t5
```

to set the timer interrupt to five seconds rather than the default of 45 seconds. This only changes the timer interrupt and leaves the rest of the print stream settings as they were.

The **printer** command issued with no options displays the current **printer** setting and options.

Flags

/d lpt*n* Deletes any buffered printer output for the specified print stream. If you omit the print stream specifier the system deletes all buffered printer output for all print streams.

/p lpt*n* Prints any buffered printer output for the specified print stream . If you omit the print stream specifier the system prints all buffered printer output for all print streams.

/r Resets the *printprogram* parameter to the default print handler.

/t*timeout* Sets a timer interrupt to close the print stream after a specified number of seconds without print request activity. You can specify *timeout* values from 1 to 3600. When you specify the **/t** flag without a *timeout* parameter, the system uses the default 45 seconds.

/t0 Removes timer interrupt, that is, printing is not initiated by timer interrupt.

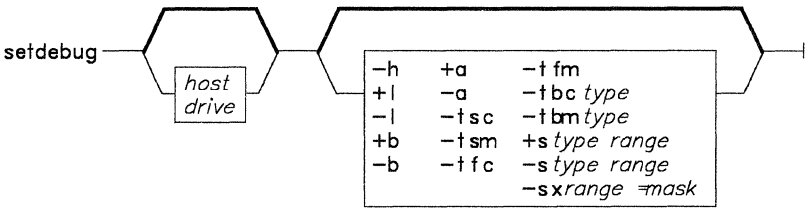
/x*n* Controls whether or not buffered print requests are processed when the print stream is closed. The default value, **/x1**, causes the system to print buffered requests when the print stream is closed. When you enter host file services, the default value is **/x1**. Specify the **/x0** option to prevent buffered print requests from being processed until explicitly requested.

setdebug Command

Purpose

Traces the DOS system call activity of your personal computer while running an Access program file services session.

Format



A5ACB015

Description

The **setdebug** command controls and displays the Access program xdebug (extended debugging) status. The command runs as an application program on DOS. Only a user with root authority can run the **setdebug** command.

Note: The Access program performance can be quite slow after running **setdebug** while the system is reporting system call activity.

The **setdebug** command manipulates the following parameters that control extended debug messages:

- xdebug error logging on/off
- Host on which the log is written
- Dump data before system calls
- Dump data after system calls
- Which system call information is logged.

The **setdebug** command causes log messages describing the system call activity on the personal computer to be written in a log file created on the specified host. The log file is called **/usr/spool/pcilog/dossvr.***xxxx*, where *xxxx* is the process identification of the DOS server for that virtual drive. If you do not specify the host on which to create the log file when starting **setdebug**, **setdebug** creates the log file on the virtual drive. Since **setdebug** logs error messages on a remote host, you must establish a virtual drive before running **setdebug**.

Notes:

1. Do not delete the log file until you are finished using **setdebug**. If you delete the log file, you must log off the Access program and log back in to begin logging system calls again.
2. Before logging off, be sure all error logging is turned off.

There are six types of extended debugging log messages that display the information available before and after DOS system calls:

- System call name
- Error code returned
- Registers

- Stack
- File control block
- ASCIIZ or $-terminating string.

The creation and transmission control of the log message is accomplished by two control tables:

MASTER table
Lists all 8 bits for each system call. If each bit for a system call is set to 1 in the MASTER table, the system call can be set. The MASTER table is contained in the **setdebug** program and cannot be changed.

CURRENT table
Used to decide whether or not to create and transmit the messages.

You can change the CURRENT table using **setdebug**. If **setdebug** tries to set a bit in the CURRENT table for an entry that does not have the corresponding bit set in the MASTER table, the bit will not be set.

Since the log file can become large very quickly, you should use **setdebug** to change the CURRENT table so that only the necessary system calls are logged. **setdebug** remembers the settings in the CURRENT table until you restart your personal computer. When you restart the personal computer, the system default settings for the CURRENT table are restored.

Both tables have a 1-byte entry for each system call. Each bit in the entry can be 1 or 0, where 1 means send, to specify whether that message is to be created and sent to the remote log.

The MASTER and CURRENT control table entries are defined as follows:

Bit 0 Spare

Bit 1 Send the ASCIIZ or $ terminated string

Bit 2 Send the file control block

Bit 3 Send the system call name

Bit 4 Send the error code returned

Bit 5 Send the registers

Bit 6 Send the stack

Bit 7 Control xdebug for this system call.

Use this table to set the *mask* parameter described in the following section.

Note: Bits are set in the CURRENT table only if the corresponding bit is set in the MASTER table. Bit 7 (control) must be set to 1 in the CURRENT table for any messages to be sent for a system call.

All numeric input to **setdebug** is in 1- or 2-digit hexadecimal numbers. Using **setdebug**, you can combine options on the command line.

If **setdebug** is called with an undefined option, it will print an `unknown option` message followed by the Access program debugging log status as follows:

```
xdebug AIX communication                ON|OFF
xdebug logging                          ON|OFF
dump data before system calls           ON|OFF
dump data after system calls            ON|OFF
debug destination host
host_name(virtual_drive)
```

The following example shows an excerpt from a **setdebug** log file. This example log was created using **setdebug** with the **+a** and **+b** options.

```
1443     - FXN CALL (0a): Buffered Keyboard Input
1444 BEF - AX: 0a00 BX: 29ed CX: 00f9 DX: 2b20 SI: 2daf DI: 006a
1445 BEF - CS: 9d02 DS: 9d02 ES: 9d02 SS: 9d02 SP: 2fcf BP: 0000 FL: f046
1446 BEF - STK (9d02:2fcf) 0228 9d02 f046 66cc 0166 0a00 0024 0000
1447 AFT - AX: 0a0d BX: 29ed CX: 00f9 DX: 2b20 SI: 2daf DI: 006a
1448 AFT - CS: 9d02 DS: 9d02 ES: 9d02 SS: 9d02 SP: 2fcf BP: 0000 FL: f046
1446 AFT - STK (9d02:2fcf) 0228 9d02 f046 66cc 0166 00a0 0024 0000
```

You can interpret the log as follows:

- The first column contains a sequential number.
- The second column indicates whether this entry contains information reflecting the state of the system before or after the system call.
- The third column contains the name of the system call, information on the state of the registers (following the AX: and CS:), or the stack information (following the STK).

Using this log, you can track the state of your system for any system call or group of system calls.

If **setdebug** is called with no options, or is called and changes any of the parameters, it will display a header message, the status of the AIX communications for extended debugging, and the current parameter status.

The exit status of **setdebug** is 1 if an error occurs, otherwise it is 0.

Flags

-h Displays a help message including the command usage, a description of the **setdebug** command parameters, and a bit map for the entries in the control tables.

+l Turns on debug logging (system call name and errors only).

-l Turns off debug logging.

+b Logs messages before each system call (also does +l).

-b Turns off debug logging before each system call (also does -l).

+a Logs messages after each system call (also does +l).

-a Turns off debug logging after each system call (also does -l).

-tsc Displays a summary of the CURRENT xdebug control table.

-tsm Displays a summary of the MASTER xdebug control table.

-tfc Displays the full hexadecimal values of the CURRENT xdebug control table.

-tfm Displays the full hexadecimal values of the MASTER xdebug control table.

-tbc*type* Displays bits of the specified *type* of the CURRENT xdebug control table.

type= c (Control)
s (Stack)
r (Registers)
e (error)
n (system call Name)
f (File control block)
a (ASCIIZ or $-terminated string)

-tbm*type*
Displays bits of the specified *type* of the MASTER xdebug control table.

+s*type range*
Sets bits of the specified *type* on for the *range* of system calls:

range = **H1[,H2][[,H3|,H3-H4]...]**

where **H***n* is a hexadecimal number between 0 and 62.

-s*type range*
Sets bits of the specified *type* off for the *range* of the system.

-sx*range* = *mask*
Sets all the bits in the *range* of system calls to *mask*. *mask* is an 8-bit hexadecimal word whose bit values have the following meaning:

Bit 0 Spare

Bit 1 Send the ASCIIZ or $-terminated string

Bit 2 Send the File control block

Bit 3 Send the system call Name

Bit 4 Send the Error code returned

Bit 5 Send the Registers

Bit 6 Send the Stack

Bit 7 Control xdebug for this system call.

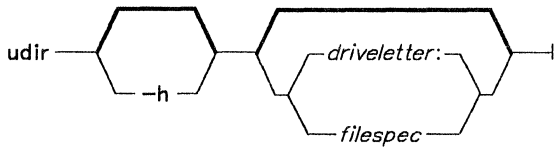
Note: Bits are set in the CURRENT table only if the corresponding bit is set in the MASTER table.

udir

Purpose

Lists the contents of directories.

Format



A5ACB016

Description

The **udir** displays the contents of directories in a combination of AIX and DOS styles, displaying both AIX and DOS mapped names, owner, file permissions, size, and modification date and time for each file for directory.

The *filespec* parameter can specify either a file or directory. Directories are indicated in the **udir** display with both an initial **d** in the permission field and the DOS <DIR> notation.

File and directory names are entered in the AIX form. The **udir** display shows all the file and directory names in both their AIX and mapped-name form. You can also use wildcard characters (* and ?), however, **udir** interprets these as DOS does, not as AIX does.

Flags

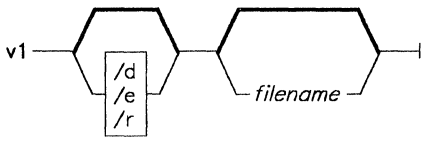
-h Displays hidden DOS files. Files on the AIX filesystem that start with a period (.), are also displayed. A dash (-) must be used for this flag.

vi

Purpose

Invokes the Access program **vi** editor.

Format



A5ACB017

Description

The Access program **vi** editor is like the AIX **vi** editor, only it runs on your personal computer and allows you to edit files on either local or virtual drives.

The *filename* parameter specifies the file or files you want to edit. You cannot specify wildcard characters (* and ?) or metacharacters (for example, `[a-z]`) as part of the file name.

You can invoke the Access program **vi** editor with any valid AIX **vi** editor parameters. However, since the Access program **vi** editor handles recovery in a different manner, using the **/r** flag yields a diagnostic message telling you to invoke **exrecovr**.

Normally, the Access program **vi** creates the temporary working file in the **\tmp** directory on your virtual disk if you invoke **vi** from a host file services session. If you do not have a host file services session in progress, the temporary file is created in the root of your current drive. The Access program **vi** names temporary files with a **vi** or **ri** prefix instead of the AIX **vi** prefixes **Ex** and **Rx**.

To change this default, set the **HOME** environment variable in **autoexec.bat** to the fully qualified directory name and create a file in that directory called **ex.rc**. This file must contain the following line:

`set dir=`*directoryname*

You can specify most of the configuration options in the Access program **vi** editor that you can specify in AIX **vi** by including them in the **ex.rc** file. In addition to the standard options, the Access program **vi** editor provides the **dosmode** (or **dm**) configuration option, which lets you edit files in DOS format.

The following AIX **vi** configuration options are not supported by the Access program **vi**:

- The **hardtabs** option. There is no need to change these values in the Access program **vi**.
- The **timeout** option. Macros always behave as if **notimeout** is set.

The differences between the AIX **vi** editor and the Access program **vi** editor include the following:

- The Access program **vi** editor does not support the AIX **vi** filters (for example, `:1,10!sort` and `:w!`).
- The Access program **vi** editor names working files with **vi** and **ri** as prefixes. AIX **vi** uses **Ex** and **Rx** as prefixes.
- The **Ctrl-H** (erase) and **Ctrl-X** (kill) functions are fixed in the Access program **vi** editor. They cannot be redefined as in the AIX **vi** editor.
- The Access program **vi** editor provides expanded file recovery capabilities to help you recover files when your session is interrupted by a host connection failure. For more information on file recovery, see Chapter 6, "The AIX Access for DOS Users vi Editor."

- The AIX **vi** configuration options **hardtabs** and **timeout** are not supported by the Access program **vi** editor.

Since backward slashes (\) are used as quote characters by **vi**, the slash (/) is the preferred directory separator. If you use backward slashes, you must double them.

Flags

/d Enters the Access program **vi** editor in DOS mode. DOS mode assumes that the file being edited is in DOS text format and writes the file in DOS text format.

/e Enters the Access program **vi** editor in **ex** mode.

/r Issues a message warning the user to invoke **exrecovr**.

Appendix A. Installing AIX Access for DOS Users

CONTENTS

About This Appendix A-3
Installing the Asynchronous Communication Adapter A-4
Installing the Network Interface Adapter A-4
3Com EtherLink Adapter A-4
Ungermann-Bass Network Interface Adapter A-4
The IBM Token-Ring Network PC Adapter A-5
Package Types A-6
Overview of Installing AIX Access for DOS Users A-7
Two Diskette-Drive System A-8
Making Backup Diskettes on a Two Diskette-Drive System A-8
Installing AIX Access for DOS Users on a Two Diskette-Drive System A-10
One Diskette-Drive System A-13
Making Backup Diskettes on a One Diskette-Drive System A-13
Installing AIX Access for DOS Users on a One Diskette-Drive System Using a RAM Disk .. A-14
Fixed-Disk Drive A-18
Making Backup Diskettes for a Fixed-Disk Drive A-18
Installing AIX Access for DOS Users on a Fixed-Disk Drive A-20

About This Appendix

The first part of this appendix contains instructions for installing an asynchronous communication adapter. For more information on adapters, see chapter one in *AIX Access for DOS Users Administrator's Guide.*

The second part of this appendix describes how to combine the Access program files from the distribution diskette with your DOS files to create a working Access program diskette or to install the Access program on your personal computer with a fixed disk.

Installing the Asynchronous Communication Adapter

You can install an asynchronous communication adapter in any full-size system unit expansion slot of your personal computer. Follow the instructions in the technical reference manual for your personal computer or in the product manual for the adapter. An RS-232 cable connects the adapter to the computer running the AIX Operating System.

Installing the Network Interface Adapter

Although you can install a number of different adapters in your personal computer system unit, you cannot use them all at the same time. You can use an asynchronous communication adapter together with any one other network interface adapter, but you cannot use two different network interface adapters together.

3Com EtherLink Adapter

The 3Com EtherLink adapter can be installed in any full-size system unit expansion slot of the personal computer (personal computer bus only), following the instructions in *3Com EtherLink Installation Guide.*

Normally, the 3Com EtherLink adapter should be configured to use interrupt channel 3 (the COM2 port). If you cannot use interrupt channel 3, you must use interrupt channel 5 and modify the **pciinit** command as described later in this appendix.

To use the 3Com EtherLink adapter with the on board transceiver, set the transceiver select switch to BNC, and attach the thin coaxial cable to the BNC coaxial cable connector on the rear panel of the board. To use the board with an external transceiver, set the transceiver select switch to DIX and attach the Ethernet transceiver cable to the 15-pin connector on the rear panel of the board.

Ungermann-Bass Network Interface Adapter

The Ungermann-Bass NIC Model IBM PS/2 can be installed in the IBM PS/2 models 50, 60, 70, and 80. The Ungermann-Bass PC NIC Model 2274A can be installed in any full-size system unit expansion slot of an IBM PC, XT, XT286, or AT personal computer, or IBM PS/2 model 25 or 30. Before installing the board you must configure it for proper operation of the Access program by setting three jumper areas.

To find the jumper areas, place the board in front of you with the rear panel and connectors on your right.

The first area, labeled W10-13, controls the physical memory location for the Ethernet device driver. This eight-pin area is near the edge of the board nearest to you and slightly right of center. (Another way to find it is to look to the left and slightly above the large ROM chip with the Ungermann-Bass label.)

Each pair of pins, from left to right, represents an address, A15 through A18. This area should be set up as follows:

W10-13	A15	A16	A17	A18
	OFF	ON	OFF	ON

OFF means that a jumper block (a rectangular metal sleeve) does not connect the pair of pins. ON means that a jumper block connects two pins.

When set properly for the Access program, you should see (from left to right) a bare pair of pins, a jumper block around the next pair, another bare pair, and a jumper block around the last pair.

The second area, labeled W9, controls the interrupt channel used by the board. This block of pins is near the right edge of the board and slightly above the slot connecters.

Each pair of pins is labeled with an interrupt (IR) number. Normally, the board should be configured to use interrupt channel 2 (the jumper block connects the two pins labeled IR2).

The IBM PC 3278/79 Emulation Adapter and the Ungermann-Bass adapter both use interrupt channel 2. If you have an IBM PC 3278/79 Emulation Adapter in your system unit, you must configure the Ungermann-Bass adapter to use a different interrupt channel so that the two boards do not conflict. Typically, you can use interrupt channel 3 (unless you also are using your COM2 port). To change the interrupt channel, remove the jumper block from the IR2 pins and place it around the IR3 pins. If you change the interrupt channel from 2, you must also modify the **pciinit** command as described later in this appendix.

The third area, labeled W1-8, controls transceiver selection. This area is a double row of eight pins each in the upper right corner of the board.

To use the board with the on board transceiver, move all the jumper blocks to the bottom row of pins, and attach the thin coaxial cable to the circular BNC coaxial cable connector on the rear panel of the board.

To use the board with an external transceiver, move all the jumper blocks to the top row of pins and attach the Ethernet transceiver cable to the 15-pin connector on the rear panel of the board.

The IBM Token-Ring Network PC Adapter

The IBM Token-Ring Network PC Adapter can be installed in any full-size system unit expansion slot of the personal computer (personal computer bus only) following the instructions in *IBM Token-Ring Network PC Adapter Guide to Operations Manual.*

Before you install the board, you must set two banks of switches: Switch Block 1 and Switch Block 2.

Switches 1 through 6 of Switch Block 1 determine the address in the computer's memory where it will locate the adapter's ROM. Set these switches to address CC000 as shown:

1	2	3	4	5	6
OFF	ON	ON	OFF	OFF	ON

If you are installing a second adapter, set switches 1 through 6 of Switch Block 1 (of the second adapter) to address DC000 as shown:

1	2	3	4	5	6
OFF	ON	OFF	OFF	OFF	ON

Switches 7 and 8 of Switch Block 1 allow you to set the interrupt level for the adapter. This is normally set to interrupt level 2. If this conflicts with another feature in your system, you can use level 3 or level 7. If you use an interrupt other than level 2, you must also modify the **pciinit** command as described later in this appendix.

If you do not have an IBM PC 3278/79 Emulation Adapter or an IBM Token-Ring Network PC Adapter already in your personal computer, set switches 7 and 8 to interrupt level 2 as shown:

```
7     8
ON    ON
```

If you cannot use interrupt level 2 and you do not have any of the following IBM personal computer adapters: PC Network, Asynchronous Communications, Synchronous Data Link Control (SDLC), or Binary Synchronous Communications (BSC); set switches 7 and 8 to interrupt level 3 as shown:

```
7     8
ON    OFF
```

If you cannot use interrupt level 2 or 3, and only if your printer does not use interrupt 7, set switches 7 and 8 to interrupt level 7 as shown:

```
7     8
OFF   OFF
```

Switch 2 of Switch Block 2 is used to tell the system unit if the adapter board is the primary or alternate board. The rest of the switches of Switch Block 2 are set by the manufacturer and should not be changed. If the adapter is the only IBM Token-Ring Network PC Adapter, set switch 2 to primary (OFF).

If you are installing a second IBM Token-Ring Network PC Adapter, set switch 2 to alternate (ON).

Package Types

During the installation process you have the option of of selecting one of three package types:

1. Basic
2. Standard
3. Development

The selection of one of these package types determines the capabilities of your system, and also which files are installed.

Basic The basic package includes the files required to operate host file services, terminal emulation, and remote printer support. The files to be installed are on diskette #1 of the Access program distribution disks. This includes the device drivers for your system configuration and the following executable files:

- **pciinit.exe**
- **login.exe**
- **logout.exe**
- **printer.exe**
- **em.com**
- **nty.exe.**

Standard

The standard package includes the files in the basic package and the following additional files:

- **vi.exe**
- **exrecovr.exe**
- **udir.exe**

- **on.exe**
- **kill.exe**
- **jobs.exe**
- **dos2aix.exe**
- **aix2dos.exe**
- **setdebug.exe**
- **doswhat.exe.**

Development

The development package has additional library files and include files. These files are used only for developing user specific applications with either the Microsoft C or Lattice C compilers. Refer to Appendix D, "Extended Library - PCILIB" for more information. This package includes the files in the basic and standard packages and the following additional files:

- **pcilcd.lib**
- **pcilcl.lib**
- **pcilcp.lib**
- **pcilcs.lib**
- **pcimscc.lib**
- **pcimscs.lib**
- **pcimscm.lib**
- **pcimscl.lib**
- **ipc.h**
- **memmdl.h**
- **pcilib.h.lib.**

Overview of Installing AIX Access for DOS Users

The Access program diskettes contain the programs necessary to establish a host file services session and the program you need for terminal emulation. The AIX Access program distribution diskettes do not contain the DOS files needed to start the system and load the Access program.

The Access program is copy-protected. You can use the distribution diskettes to create backup and working copies for your own use only.

This appendix describes, step by step, how to back up your AIX Access program distribution diskettes, how to create Access program working diskettes, a RAM disk, or how to install the Access program on your fixed disk.

These steps are described separately for a personal computer with two double-sided diskette drives, a personal computer with a single diskette drive, and personal computer with a fixed-disk drive.

Refer to the section of the appendix that corresponds to your personal computer configuration.

Two Diskette-Drive System

Follow these instructions to format three diskettes (one with the DOS operating system files), back up your AIX Access program distribution diskettes, then combine the Access program with DOS to create a set of working diskettes. (For further information on the commands used in this procedure, refer to your *Disk Operating System* manual).

For this procedure, you need the following:

- A personal computer with two double-sided diskette drives
- Six blank double-sided, double-density diskettes
- A DOS system diskette
- The AIX Access program distribution diskettes.

You also need the following information, usually available from the person who administers your system:

- The type of network interface adapter installed in your personal computer
- The network interface adapter's interrupt level
- Your networking environment: Token-Ring, RS-232, or Ethernet
- If you are using a LAN connection path, your personal computer's internet address.

Making Backup Diskettes on a Two Diskette-Drive System

1. Use a felt-tip pen to label three of the blank diskettes Access Program Working #1, Access Program Working #2, and Access Program Working #3. Label the other three blank diskettes Access Program Backup #1, Access Program Backup #2, and Access Program Backup #3. On all three working diskettes, indicate the type of network interface adapter installed in your personal computer. You can only use these working diskettes in personal computers containing the indicated type of network interface adapter.
2. Insert your DOS system diskette into diskette drive A.
3. Switch on your system unit if you have not already done so. If the system unit is already on, press and hold the **Ctrl** and **Alt** keys, then press the **Del** key (system reset). Release all three keys.
4. DOS asks for today's date; enter the date.
5. DOS asks for the time; enter the time.

   A message similar to the following is displayed:

   ```
   IBM Personal Computer DOS version 3.30
   (c) Copyright International Business Machines, Corp. 1981, 1987
   (c) Copyright Microsoft Corp. 1981, 1986
   ```

6. Type:

   ```
   format b: /s
   ```

7. Press **Enter**.

   The following message is displayed:

   ```
   Insert new diskette for drive B:
   and strike any key when ready
   ```

8. Insert the blank Working #1 diskette into drive B.

Press any key. When the **format** procedure is completed, a message similar to the following is displayed:

```
Formatting. . .Format complete

     xxxxxx bytes total disk space
      xxxxx bytes used by system
     xxxxxx bytes available on disk

Format another (Y/N)?
```

9. Type N and press **Enter**.
10. Remove the formatted Working #1 diskette from drive B and set it aside. You will install the Access program files later.
11. Type:

    ```
    format b:
    ```

12. Press **Enter**.

    The following message is displayed:

    ```
    Insert new diskette for drive B:
    and strike ENTER when ready
    ```

13. Insert the blank Working #2 diskette into drive B.
14. Press **Enter**.

    When the **format** procedure is completed, a message similar to the following is displayed:

    ```
    Formatting...Format complete

        xxxxxx  bytes  total disk space
         xxxxx  bytes  used by system
        xxxxxx  bytes  available on disk

    Format another (Y/N)?
    ```

15. Type Y and press **Enter**.

    The following message is displayed:

    ```
    Insert new diskette for drive B:
    and strike ENTER when ready
    ```

16. Remove the formatted Working #2 diskette from drive B and set it aside. You will install the Access program files on it later.
17. Insert the blank Working #3 diskette into drive B.
18. Press **Enter**.

    When the **format** procedure is completed, the following message is displayed:

    ```
    Formatting....Format complete

      xxxxxx  bytes  total disk space
       xxxxx  bytes  used by system
      xxxxxx  bytes  available on disk

    Format another (Y/N)?
    ```

19. Type N and press **Enter**.
20. Remove the formatted Working #3 diskette from drive B and set it aside. You will install the Access program files on it later.
21. Type:

    ```
    diskcopy a: b:
    ```

22. Press **Enter**.

The following message is displayed:

```
Insert source diskette in drive A:
Insert target diskette in drive B:
Press any key when ready
```

23. Remove the DOS system diskette from drive A and store it in a safe place.
24. Insert AIX Access program distribution diskette #1 into drive A. Insert the blank Backup #1 diskette into drive B.
25. Press any key. The *in use* lights alternately come on while the AIX Access program distribution diskette is being copied.

    When the copying is complete, the following message is displayed on your screen:

    ```
    Copy another diskette (Y/N)?
    ```

26. Press `Y`. The following message is displayed:

    ```
    Insert source diskette in drive A:
    Insert target diskette in drive B:
    Press any key when ready
    ```

27. Remove the AIX Access program distribution diskette #1 from drive A and store it in a safe place.
28. Remove the Backup #1 diskette from drive B and set it aside for now.
29. Repeat steps 24 through 28 for distribution diskettes #2 and #3 and the corresponding backup diskettes, except that for diskette #3, respond `n` at step 26 (`copy another diskette (y/n)?`).
30. Insert the Working #1 diskette into drive A.
31. Press any key. The DOS prompt `A>` is displayed.

    Your Access program backup is complete. You now use the Backup #1, Backup #2, and Backup #3 diskettes for the remainder of the installation procedure.

Installing AIX Access for DOS Users on a Two Diskette-Drive System

1. Insert your Access Program Backup #1 diskette into drive B.
2. Type `b:` and press **Enter**.
3. Enter the following:

   ```
   install
   ```

   A message similar to the following is displayed:

   ```
   Install, (c) Copyright International Business
                Machines Corporation 1988
            (c) Copyright Locus Computing Corporation 1984, 1988

   What drive do you want to copy to (default A:)?
   ```

4. Type `A` and press **Enter**.

   A message similar to the following is displayed:

```
Package types available:
    (1) Basic (minimum necessary for AIX Access for DOS Users)
    (2) Standard (basic + utility programs)
    (3) Development (standard + libraries and include files)
Please enter the number for the desired package type.
```

5. Type the number corresponding to the installation you want[1] and press **Enter**.

   The following message is displayed listing the network types available:

```
Network types available:
    (1) Ethernet and RS-232 Interface
    (2) IEEE 802 Ethernet and RS-232 Interface
    (3) Token Ring and RS-232 Interface
    (4) IEEE 802 Token Ring and RS-232 Interface
    (5) RS-232 Interface Only
Enter the number corresponding to your network type:
```

6. Choose the appropriate LAN environment (or `RS-232 Interface Only` if your personal computer does not have a LAN connection), and type the corresponding number.

   Press **Enter**.

7. If you selected option 5, go to step 9.

   If you selected option 1 or 2, the following message is displayed:

```
Network interface devices available:
    (1) Ungermann-Bass PC NIC
    (2) 3Com 3C501
Enter the number corresponding to your network interface:
```

   If you selected option 3 or 4 in step 6, the following message is displayed:

```
Network interface devices available:
    (1) IBM Token Ring Adapter
Enter the number corresponding to your network interface:
```

8. Choose the appropriate network interface adapter, and type the corresponding number.

   Press **Enter**.

   The following message is displayed:

```
Enter the Internet Address of your PC in the form A.B.C.D:
```

   Type your internet address and press **Enter**.

9. A message similar to the following is displayed:

```
Do you want to install the Access program
utilities in a subdirectory? [y]
```

10. If you want the Access program utilities in a subdirectory, press `Y`:

---

[1] Most of the examples in this manual assume that you have chosen the standard installation.

If you want the Access program utilities in the **root** directory on drive A, type:

```
n
```

Press **Enter**. If you typed n, proceed to 13.

11. A prompt for a path name for the subdirectory is displayed:

    ```
    Enter path: C:\
    ```

12. Enter the full path name of the subdirectory in which you want the Access program utilities installed.

    Press **Enter**.

13. Your selections for machine type, installation type, network type, network interface device, and internet address, if appropriate, are displayed. The following message is then displayed:

    ```
    Is this information correct (Y,N,Q)? [Q]
    ```

14. You have the opportunity to re-enter any information by selecting n or terminating the installation by selecting q. If you select n, **install** takes you back to step 5. If all the information is correct, select y.

15. Press **Enter**. A list of the Access program files being copied to your Working #1 diskette is displayed.

16. When all the files for this diskette have been copied, a message similar to the following is displayed:

    ```
    Replace the AIX Access program distribution diskette
     with diskette number 2
    Press any key when ready
    ```

17. Remove the Backup #1 diskette from drive B and store it in a safe place. Remove the Working #1 diskette from drive A and set it aside.

18. Insert the Working #2 diskette in drive A and the Backup #2 diskette in drive B.

19. Press any key.

    A list of the Access program files being copied to your Working #2 diskette is displayed.[2]

20. When all the files for this diskette have been copied, a message similar to the following is displayed:

    ```
    Replace the AIX Access program distribution diskette
     with diskette number 3
    Press any key when ready
    ```

21. Remove the Backup #2 diskette from drive B and store it in a safe place. Remove the Working #2 diskette from drive A and set it aside.

22. Insert the Working #3 diskette in drive A and the Backup #3 diskette in drive B.

23. Press any key. A list of the Access program files being copied to your Working #3 diskette is displayed.[3]

24. When all the files are copied to the Working #3 diskette, the following message is displayed:

    ```
    AIX Access for DOS Users installation complete
    ```

---

2 If the Working #2 diskette is not in the drive, **install** informs you have the wrong diskette in the drive and takes you back to step 16.

3 If the Working #3 diskette is not in the drive, **install** informs you that you have the wrong diskette in the drive and takes you back to step 20.

25. Remove the Backup #3 diskette from drive B and store it in a safe place. Remove the Working #3 diskette from drive A and set it aside.

26. If your network interface adapter is configured to use an interrupt channel other than the default interrupt channel, edit the **autoexec.bat** file, adding the following option to the end of the **pciinit** command line:

    `-x`*n*

    where *n* is the interrupt channel plus 8.

    Your Access Program Working diskettes are ready to use. If you want to proceed with the Access program session, insert the Working #1 diskette into drive A, hold down the **Ctrl** and **Alt** keys and press the **Del** key to reset your personal computer and initialize the Access program.

One Diskette-Drive System

Follow these instructions to create a RAM disk, back up your AIX Access program distribution diskettes, and combine the Access program with DOS on your RAM disk. For further information on the commands used in this procedure, refer to *AIX Operating System Technical Reference.*

For this procedure, you need the following:

- A personal computer with one double-sided diskette drive and 512KB of internal memory
- Four blank double-sided, double-density diskettes
- A DOS system diskette
- The AIX Access program distribution diskettes.

You also need the following information, usually available from the person who administers your system:

- The type of network: Token-Ring, RS-232, or Ethernet
- The type of network interface adapter installed in your personal computer
- The network interface adapter's interrupt level or IRQ level if you are using Token-Ring or Ethernet
- Your personal computer's internet address if you are using Token-Ring or Ethernet.

Making Backup Diskettes on a One Diskette-Drive System

1. Use a felt-tip pen to label the blank diskettes Access Program Backup #1, Access Program Backup #2, and Access Program Backup #3.
2. Insert the DOS system diskette into the diskette drive and switch on your system unit if you have not already done so. If the system unit is already on, press and hold the **Ctrl** and **Alt** keys, then press the **Del** key. Release all three keys.
3. DOS asks for today's date; enter the date.
4. DOS asks for the time; enter the time.

   A message similar to the following is displayed:

   ```
   IBM Personal Computer DOS version 3.30
   (c) Copyright International Business Machines Corp. 1981, 1987
   (c) Copyright Microsoft Corp. 1981, 1986
   ```

5. Enter:

   ```
   diskcopy a: a:
   ```

The following message is displayed:

```
Insert source diskette into drive A:

Press any key when ready
```

Note: During the diskcopy procedure, the term **SOURCE** refers to the distribution diskettes and **TARGET** refers to the backup diskettes.

6. Remove the DOS system dsikette from drive A, and insert the AIX Access program distribution diskette #1 into drive A.

   Press any key. The *in use* light comes on while the Access program distribution diskette is read; the following message is then displayed:

   ```
   Insert target diskette into drive A:

   Press any key when ready
   ```

7. Remove your AIX Access program distribution diskette #1 and insert the blank Backup #1 diskette.
8. Press any key.

   With only one diskette drive, you must exchange diskettes during the **diskcopy** procedure. A message is displayed on your screen when an exchange is needed.

   Exchange diskettes when requested until the following message is displayed:

   ```
   Copy another diskette (Y/N)?
   ```

9. Press **Y**.

   The following message is displayed:

   ```
   Insert source diskette into drive A:

   Press any key when ready
   ```

10. Remove the Backup #1 diskette from drive A and set it aside for now. Insert the AIX Access program distribution diskette #2 into drive A.

    Press any key.

11. Repeat steps 7 through 9 for distribution diskettes #2 and #3, except that at step 9 for distribution diskette #3, press **N** instead of **Y**.

    The following message is displayed:

    ```
    Insert disk with COMMAND.COM in drive A
    and strike any key when ready
    ```

12. Remove the Backup #3 diskette from drive A and set it aside. Insert the DOS system diskette in drive A and press any key.

Your Access program backup is complete. Store your original AIX Access program distribution diskettes in a safe place, and use the backup diskettes for the rest of the installation procedure.

Installing AIX Access for DOS Users on a One Diskette-Drive System Using a RAM Disk

Follow these instructions after creating the backup diskettes to install AIX Access for DOS Users on a system that contains only a single diskette drive and no other mass storage devices.

1. Label a blank diskette `DOS System with ACCESS`. This will become the system diskette used to start your system and initialize the Access program
2. Insert an IBM DOS system diskette in drive A, and enter:

```
format  a: /s
```

The following message is displayed:

```
Insert new diskette for drive A:
and strike ENTER when ready
```

3. Remove the IBM DOS diskette, insert the diskette labeled DOS System with ACCESS, and press **Enter.**.

   When the **format** procedure is completed, the following message is displayed:

```
Format Complete
System transferred
    xxxxxx  bytes total disk space
     xxxxx  bytes used by system
    xxxxxx  bytes available on disk
Format another (Y/N)?
```

4. Type **N** and press **Enter**
5. Create a **config.sys** file by entering the following:

```
copy con config.sys
```

6. Type the following and then press **F6**, followed by **Enter**:

```
device = vdisk.sys 310
```

7. Remove the diskette labeled DOS System with ACCESS and insert the IBM DOS system diskette. Then enter the following:

```
copy vdisk.sys B:
```

   The computer responds with this message:

```
Insert diskette for drive B: and strike
any key when ready
```

8. Remove the IBM DOS system diskette, reinsert the DOS System with ACCESS diskette, and press any key.
9. After the copy process has completed, restart the computer by pressing **Ctrl-Alt-Del**. The computer will restart and create a RAM Disk as drive C:.
10. At the date prompt, enter the date.
11. At the time prompt, enter the time.
12. Remove the DOS System with ACCESS diskette and insert your Access Program Backup #1 diskette. Type a and press **Enter**.
13. Type:

```
install
```

    Press **Enter**.

    A message similar to the following is displayed:

```
Install, (c) Copyright International Business
          Machines Corporation 1988
         (c) Copyright Locus Computing Corporation 1984, 1988

What drive do you want to copy to (default C:)?
```

14. Press **Enter**. A message similar to the following is displayed:

```
Package types available:
    (1) Basic (minimum necessary for AIX Access for DOS Users)
    (2) Standard (basic + utility programs)
    (3) Development (standard + libraries and include files)
Please enter the number for the desired package type.
```

15. Only the Basic package can be installed on a single diskette system, so type 1 and press **Enter**.

    The following message is displayed listing the available network types:

```
Network types available:
    (1) Ethernet and RS-232 Interface
    (2) IEEE 802 Ethernet and RS-232 Interface
    (3) Token Ring and RS-232 Interface
    (4) IEEE 802 Token Ring and RS-232 Interface
    (5) RS-232 Interface Only
Enter the number corresponding to your network type:
```

16. Choose the appropriate LAN environment (or RS-232 Interface Only if your personal computer does not have a LAN connection) and type the corresponding number.

17. If you selected option 5, proceed to step 21.

    If you selected option 1 or 2, the following message is displayed:

```
Network interface devices available:
    (1) Ungermann-Bass PC NIC
    (2) 3Com 3C501
Enter the number corresponding to your network interface:
```

    If you selected option 3 or 4 in step 16, the following message is displayed:

```
Network interface devices available:
    (1) IBM Token Ring Adapter
Enter the number corresponding to your network interface:
```

18. Choose the appropriate network interface adapter and enter the corresponding number. The following message is displayed:

```
Enter the IRQ level (default 2)
```

19. Enter the IRQ level (interrupt level) for your LAN adapter. This information may be obtained from the person who administers your system.

    The following message is displayed:

```
Enter the internet address of your PC in the form A.B.C.D:
```

20. Type your internet address and press **Enter**. This information may be obtained from the person who administers your system.

21. A message similar to the following is displayed:

```
Do you want to install the Access program
utilities in a subdirectory?[y]
```

22. Type n and press **Enter**.

23. Your selections for machine type, installation type, network type, network interface device, and internet address, if appropriate, are displayed. The following message then is displayed:

```
Is this information correct (Y,N,Q)? [Q]
```

24. You have the opportunity to re-enter any information by selecting `n` or terminating the installation by selecting `q`. If you select `n`, **install** takes you back to step 15. If all information is correct, select `y`.

    Press **Enter**. A list of the Access program files being copied from the Backup #1 diskette is displayed. Note that **install** copies any existing **autoexec.bat** and **config.sys** files to **ae.bat** and **cf.sys**, respectively, and creates (if none existed) or modifies **autoexec.bat** and **config.sys**.

25. When all the files for this diskette have been copied, a message similar to the following is displayed:

```
Replace the AIX Access program distribution diskette
with diskette number 2
Press any key when ready
```

    Remove the Backup #1 diskette and store it in a safe place.

26. Insert the Backup #2 diskette and press any key.

    A list of the Access program files being copied from the Backup #2 diskette is displayed.[4]

27. When all the files for this diskette have been copied, a message similar to the following is displayed:

```
Replace the AIX Access program distribution diskette
with diskette number 3
Press any key when ready
```

28. Remove the Backup #2 diskette and store it in a safe place.

29. Insert the Backup #3 diskette and press any key.

    A list of the Access program files being copied from your Backup #3 diskette is displayed.[5]

30. When all the files for this diskette have been copied, the following message is displayed:

```
AIX Access for DOS Users installation complete
```

31. Remove the Backup #3 diskette and reinsert the Backup #2 diskette.

32. The **config.sys** file must be modified to delete any references to the C drive. Enter the following command to edit the **config.sys** file:

```
vi c:config.sys
```

    The display of the **config.sys** file will be similar to the following:

```
device = C:\mtd.sys
device = C:\bridge.sys -D:4
```

    Delete all occurrences of **C:** so that the C drive designation is omitted. Refer to Chapter 6, "The AIX Access for DOS Users vi Editor" if you are unfamiliar with the vi editor.

33. Exit from **vi**. The following message is displayed:

---

[4] If the Backup #2 diskette is not in the drive, **install** informs you have the wrong diskette in the drive and takes you back to step 25.

[5] If the Backup #3 diskette is not in the drive, **install** informs you that you have the wrong diskette in the drive and takes you back to step 27.

```
Insert disk with COMMAND.COM in drive A
and strike any key when ready
```

34. Remove the Access program Backup #2 diskette, insert the DOS System with ACCESS diskette and press any key.
35. Copy all files from the RAM disk (referred to as C:) to the A drive with the following command:

```
copy c:*.* a:
```

The Access program is ready to use. If you want to proceed with an AIX Access program session, restart the computer and initialize the Access program by pressing **Ctrl-Alt-Del**.

Fixed-Disk Drive

Follow these instructions to back up your AIX Access program distribution diskettes and combine the Access program with DOS on your fixed disk. (For further information on the commands used in this procedure, refer to your *Disk Operating System* manual.)

It is assumed that you have set up a DOS partition on your fixed disk and have copied DOS into that partition.

For this procedure, you need the following:

- A personal computer with at least one double-sided diskette drive and a fixed disk
- Three blank double-sided, double-density diskettes
- Your AIX Access program distribution diskettes.

You also need the following information, usually available from the person who administers your system:

- The type of network interface adapter installed in your personal computer
- Your networking environment: Token-Ring, Ethernet, or RS-232 only
- If you are using a LAN connection path, your personal computer's internet address.

Making Backup Diskettes for a Fixed-Disk Drive

1. Use a felt-tip pen and label the blank diskettes Access Program Backup #1, Access Program Backup #2, and Access Program Backup #3.
2. Keeping the door open on diskette drive A, switch on your system unit if you have not already done so. If the system unit is already on, press and hold the **Ctrl** and **Alt** keys, then press the **Del** key. Release all three keys.

   Note: Steps 1 through 4 bring up the DOS prompt. If your existing **autoexec.bat** file causes your system to load some program other than DOS, bring up the DOS prompt C> and continue with step 5.
3. DOS asks for today's date; enter the date.
4. DOS asks for the time; enter the time.

A message similar to the following is displayed:

```
IBM Personal Computer DOS version 3.30
(c) Copyright International Business Machines Corp. 1981, 1987
(c) Copyright Microsoft Corp. 1981, 1986
```

5. Type:

   ```
   diskcopy a: a:
   ```

   Press **Enter**.

   The following message is displayed:

   ```
   Insert source diskette into drive A:
   Press any key when ready
   ```

6. Insert the AIX Access program distribution diskette #1 into drive A. For this procedure, use the AIX Access program distribution diskette as your source diskette.

   With only one diskette drive, you must exchange diskettes during the **diskcopy** procedure. A message is displayed on your screen when an exchange is needed.

   Press any key. The *in use* light comes on while the Access program distribution diskette is read; the following message is then displayed:

   ```
   Insert target diskette into drive A:
   Press any key when ready
   ```

7. Remove your AIX Access program distribution diskette #1 and insert the blank Backup #1 diskette. Use the Backup #1 diskette as the target diskette.

8. Press any key.

   The *in use* light comes on while the backup diskette is written. The screen continues to ask for the source and target diskettes until the copy is complete. Follow the messages on your screen until the following message is displayed:

   ```
   Copy another diskette (Y/N)?
   ```

9. Type:

   ```
   y
   ```

   The following message is displayed:

   ```
   Insert source diskette into drive A:
   Press any key when ready
   ```

10. Remove the Backup #1 diskette from drive A and set it aside for now. Insert the AIX Access program distribution diskette #2 into drive A.

    Press any key.

11. Repeat steps 7 through 9 for distribution diskettes 2 and 3, except at step 9 for distribution diskette 3, type `n`.

12. Remove the Backup #3 diskette from drive A and set it aside.

Your Access program backup is complete. Store your original AIX Access program distribution diskettes in a safe place, and use the backup diskette for the rest of the installation procedure.

Installing AIX Access for DOS Users on a Fixed-Disk Drive

1. Insert your Access Program Backup #1 diskette into drive A, and type:

   ```
   a:
   ```

   Press **Enter**.

2. Type:

   ```
   install
   ```

   Press **Enter**.

   A message similar to the following is displayed:

   ```
   Install, (c) Copyright International Business
                Machines Corporation 1988
            (c) Copyright Locus Computing Corporation 1984, 1988

   What drive do you want to copy to (default C:)?
   ```

3. Type

   ```
   c:
   ```

   Press **Enter**.

   A message similar to the following is displayed:

   ```
   Package types available:
       (1) Basic (minimum necessary for AIX Access for DOS Users)
       (2) Standard (basic + utility programs)
       (3) Development (standard + libraries and include files)
   Please enter the number for the desired package type.
   ```

4. Type the number corresponding to the installation you want and press **Enter**.

   The following message is displayed listing the network types available:

   ```
   Network types available:
       (1) Ethernet and RS-232 Interface
       (2) IEEE 802 Ethernet and RS-232 Interface
       (3) Token Ring and RS-232 Interface
       (4) IEEE 802 Token Ring and RS-232 Interface
       (5) RS-232 Interface Only
   Enter the number corresponding to your network type:
   ```

5. Choose the appropriate LAN environment (or `RS-232 Interface Only` if your personal computer does not have a LAN connection) and type the corresponding number.

   Press **Enter**.

6. If you selected option 5, proceed to step 10.

   If you selected option 1 or 2, the following message is displayed:

   ```
   Network interface devices available:
       (1) Ungermann-Bass PC NIC
       (2) 3Com 3C501
   Enter the number corresponding to your network interface:
   ```

   If you selected option 3 or 4 in step 5, the following message is displayed:

```
Network interface devices available:
    (1) IBM Token Ring Adapter
Enter the number corresponding to your network interface:
```

7. Choose the appropriate network interface adapter and type corresponding number.
8. Press **Enter**.

   If you selected a LAN in step 5, the following message is displayed:

   ```
   Enter the internet address of your PC in the form A.B.C.D:
   ```

9. Type your internet address and press **Enter**.
10. The following message is displayed:

    ```
    Do you want to install the Access program
    utilities in a subdirectory?[y]
    ```

11. If you want the Access program utilities in a subdirectory, type:

    ```
    y
    ```

    If you want the Access program utilities in the **root** directory on drive C, type:

    ```
    n
    ```

    Press **Enter**. If you typed n, proceed to step 14.
12. A prompt for a path name for the subdirectory is displayed:

    ```
    Enter path: C:\
    ```

13. Enter the full path name of the subdirectory in which you want the Access program utilities installed.

    Press **Enter**.
14. Your selections for machine type, installation type, network type, network interface device, and internet address, if appropriate, are displayed. The following message is then displayed:

    ```
    Is this information correct (Y,N,Q)? [Q]
    ```

15. You have the opportunity to re-enter any information by selecting n or terminating the installation by selecting q. If you select n, **install** takes you back to step 4. If all information is correct, select y.

    Press **Enter**. A list of the Access program files being copied from the Backup #1 diskette is displayed. Note that **install** copies any existing **autoexec.bat** and **config.sys** files to **ae.bat** and **cf.bat**, respectively, and creates (if none existed) or modifies **autoexec.bat** and **config.sys**.
16. When all the files from the backup diskette are copied, a message similar to the following is displayed:

    ```
    Replace AIX Access program distribution diskette with diskette # 2
    Press any key when ready
    ```

17. Remove the Backup #1 diskette from drive A and store it in a safe place.
18. Insert the Backup #2 diskette into drive A, and press any key.
19. Repeat steps 16 through 18 for the Backup #3 diskette.
20. When all files from the Backup #3 diskette are copied, the following message is displayed:

    ```
    AIX Access for DOS Users installation complete.
    ```

    If you installed the Access program in a subdirectory, add that subdirectory to the *path* variable in **autoexec.bat** file. Be sure that the current drive is the one where you installed the package described in step 3.

21. Remove the Backup #3 diskette from drive A and store it in a safe place.
22. If your network interface adapter is configured to use an interrupt channel other than the default interrupt channel, edit the **autoexec.bat** file, adding the following option to the end of the **pciinit** command line:

    -x*n*

    where *n* is the interrupt channel plus 8.

The Access program is ready to use. If you want to proceed with an AIX Access program session, open the door of drive A, hold down the **Ctrl** and **Alt** keys, press **Del** key to reset your personal computer and initialize the Access program.

Appendix B. Messages

This appendix lists messages that the Access program can generate. These messages appear in alphabetical order to facilitate your locating the message you want. Each message entry includes the exact wording of the message and a complete description including the cause and recommended user response.

The Access program **vi** utility error messages are listed in a separate section at the end of this appendix.

This appendix does not list DOS messages. If your message does not appear in this appendix, check your DOS manual or contact the person who administers your system.

a-abort, c-continue, d-detach:

Cause: You pressed **Ctrl-C** or **Ctrl-Break** while **on** was running in the DOS foreground.

Action: Typing `a` stops the **on** command. Typing `c` continues running it as before. Typing `d` places the requested AIX command in the background.

Access Denied

Cause: The specified file is inaccessible for one of the following reasons:

- The user invoking it does not own the file.
- A directory on the path to the file is not executable.
- The file is locked (that is, it is already opened by an application that supports file sharing).

Action: Check that the directory containing the file and all directories on the path to the file have execute permission. If the file is locked, wait until it is closed and the lock is removed.

aixcommand: **access denied or file not found**

Cause: A program file in the pipeline of the requested AIX command was not located, or you do not have execute permission to use it. The field *aixcommand* is replaced with the name of the first requested AIX command in the **on** command line.

Action: Check to make sure that you typed the name of each command correctly, that each command is located in a directory named in your **UPATH** setting, and that you have execute permission for each named command.

Access program not initialized

Cause: The **login** command was invoked before the Access program was initialized.

Action: Initialize the Access program using the **pciinit** command, then retry the **login** command.

Access program session terminated. Login to retry.

Cause: You answered n to the `Error -- Host did not respond Retry?` message. If you choose not to retry, your session is terminated.

Action: No action is required. If you want to reestablish host file services to that host, begin by typing the **login** command to start a new session to that host.

AIX Access for DOS Users Copy Protection Violation - SYSTEM DISABLED

Cause: Two personal computer users attempted to use copies of the same AIX Access program distribution diskette to log in to the Access program simultaneously. The first user to log in is unaffected. The second user attempting to log in receives this message.

Action: The disabled personal computer must be powered off and powered on again to be usable. Do not attempt to log in to the Access program using a duplicate disk. Simultaneous Access program sessions can be established only by using a unique working disk for each session. Contact the person who administers your system.

AIX exec failed

Cause: on was unable to initiate the requested AIX command on the host for a reason other than inability to access the requested program files. This message is generated when the maximum-processes-per-user-limit is reached on the host, among other reasons.

Action: Use **on** - ps to determine if the maximum processes limit has been reached. If so, wait until some processes finish before initiating others. If not, consult the person who administers your network for other possible failures on the host.

An error has occurred while reading the input file

Cause: The source file is corrupted or there was an operating system error.

Action: Check the integrity of the source file. If it does not appear to be corrupted, retry the operation. If the error occurs again, consult the person who administers your system.

An error has occurred while writing output file

Cause: This message can be caused by any of the following:

- Insufficient disk space to accommodate the target file
- Incorrect AIX permission mode for the target directory
- An operating system error.

Action: Verify whether there is room on your disk or diskette for the target file. If necessary, delete unneeded files. If there is adequate space for the target file, check the AIX permission mode of the target directory and any directories on the path to the target directory. The target directory must be writable and executable, and directories on the path to the target directory must be executable. If these measures do not solve the problem, consult the person who administers your system.

Can't open input file

Cause: The source file does not exist or is not accessible.

Action: Check that the source file exists and is named correctly. Check that the file is readable. Check that the directory containing the source file and any other directories on the path to the source file are executable.

Can't open output file

Cause: The target file cannot be opened, because the directory containing it is not executable and writable, because a directory on the path to the target directory is not executable, or because the target file already exists and is write-protected.

Action: Check that the target directory is writable and executable. Check that all directories on the path to the target directory are executable. Check whether a file already exists with the name of the target file. If you intend to replace an existing file with the new file created by **dos2aix** or **aix2dos**, the existing file must have write permission. Instead of replacing an existing file, you can choose a different name for the **dos2aix** or :hp2aix2dos target file.

Cannot find correct BRIDGE device

Cause: The Access program device driver was not successfully loaded.

Action: Check that your Access program working disk contains the **bridge.sys** and **config.sys** files. If either file is missing, reinstall the Access program from the distribution diskettes. Check that the **config.sys** file properly specifies the location of **bridge.sys**. Retry the operation.

Cannot open session, error 3

Cause: This message is displayed after the list of available hosts has been displayed and the user selects a host for an emulation session. The host might have gone down between the time the list was displayed and the time the user selected that host. There might also be a communication problem between the personal computer and the host.

Action: Retry the operation. If the problem persists, consult the person who administers your system.

DOS 3.3 is required.

Cause: The **on** utilities require DOS 3.3 on your personal computer.

Action: Upgrade your personal computer operating system to DOS 3.3.

DOS memory allocation error

Cause: An error occurred when **on** attempted to reserve a portion of DOS memory to hold this request.

Action: Restart your personal computer and try again. If the problem persists, contact your Access program supplier.

Error-Host did not respond Retry? (Y or N)

Cause: There are three reasons why this message can be displayed:

1. The host might have responded too slowly to a file services request or to an **F10** return to host file services from terminal emulation.
2. The host might have gone down between the time when you selected that host from the host file services Host-selection menu and the time your connection request was transmitted.
3. The host might have gone down and come back up while you were idle or working in emulation, thereby breaking your underlying host file services connection.

Action: If you are sure the host has gone down, enter n; you are returned to DOS. If you think the host might be up, enter y and another connection request will be transmitted.

Error in initialization

Cause: The Access program device driver was not successfully loaded.

Action: Check that your Access program working disk contains the **bridge.sys** and **config.sys** files. If either file is missing, reinstall the Access program from the distribution diskette. Check that the **config.sys** file properly specifies the location of **bridge.sys**. Retry the operation.

Error in network service.

Cause: A hardware or software error occurred in the LAN or RS-232 connection to the host.

Action: Confirm with the person who administers your network that network services are available at this time. If so, check all hardware and wiring connections.

File not found

Cause: The specified file is inaccessible for one of the following reasons:

- The user invoking it does not own the file.
- A directory on the path to the file is not executable.
- The file is locked (that is, it is already opened by an application that supports file sharing).

Action: Check that you own the file. Check that the directory containing the file and all directories on the path to the file have execute permission. If the file is locked, you must wait until it is closed and the lock is removed.

aixcommand: **host name is not a connected host**

Cause: You did not initiate host file services before attempting to run **on**.

Action: Use the Access program **login** command to initiate host file services.

Invalid combination of options

Cause: The **dos2aix** and **aix2dos** options are combined in an illegal way. The **/b**, **/u**, and **/l** options are mutually exclusive.

Action: Retry the operation with a legal combination of options.

Invalid format

Cause: **on** could not execute the command as typed.

Action: Check your **on** command for logical consistency, especially as regards DOS and AIX pipes and redirection, and type a corrected command.

jobs: bad numeric argument (*nn*)

Cause: The job number (*nn*) you typed was not found in the job table.

Action: Retype a corrected command.

jobs: job not found

Cause: You attempted to reattach to a detached task with an invalid job ID number.

Action: Display the current job table with the **jobs** command. Retype your attachment request with the correct job ID number.

aixcommand: **job table full**

Cause: When you tried to run the *aixcommand* with **on**, the job table was filled to capacity with the record of **on**-initiated tasks.

Action: Clear completed jobs from the **jobs** command.

kill: Must specify job or process id.

Cause: **kill** was entered without designating the job you want to terminate.

Action: Retype a corrected command.

hostname **Logged Out**

Cause: This message is displayed whenever the **logout** command is issued.

Action: None.

Login Aborted

Cause: This message is displayed when you abort the login process over an RS-232 connection path by pressing **F10** or when you answer `no` to the `Login Incorrect. Try again?` prompt. The message acknowledges that the process has been aborted, as requested by the user.

Action: None.

Login Incorrect. Try Again? (y or n)

Cause: Your user name or password was entered incorrectly.

Action: Type y or n. If you choose **y** to retry the login operation, type your user name and password carefully. If your login is still unsuccessful, consult the person who administers your system.

No Hosts Available For Service

Cause: No response was received to the request for a host map and there are no service ports on this personal computer.

Action: This is a host system problem. Consult the person who administers your system.

ON: *driveletter* **is not a virtual drive.**

Cause: A drive letter was specified for which there is no active file services session.

Action: Reenter the command with an active file services driveletter.

ON: *hostname* **is not a connected host**

Cause: A host was specified with an **on** command that is currently not connected to your system.

Action: Re-enter the command with an active file services host specified.

Parameter Error

Cause: An incorrect entry was made on the **bridge.sys** line in **config.sys**.

Action: Make sure that any changes you made to your **config.sys** file are valid. Refer to "Setting a Default Search Path" and "Setting the on Command Variables" for more information.

Path Not Found

Cause: The path specifying the location of the file does not exist.

Action: Verify the location of the file and enter the path in the correct form.

PCIINIT: Bad Argument

Cause: pciinit options are specified incorrectly. **pciinit** requires its options to be specified with the AIX switch character (-) rather than the normal DOS switch character (/).

Action: Retry the **pciinit** command specifying the options with a hyphen (-) and in lowercase.

PCIINIT: Unknown switch

Cause: An invalid **pciinit** option is specified.

Action: Read the description of **pciinit** in this book and retry the operation with valid options.

Selected Baud Rate Invalid -- Try Again

Cause: An invalid baud rate was specified for an RS-232 path connection.

Action: Enter a valid baud rate chosen from the baud rates shown.

Selected Host Not Available. Try Another? (y or n)

Cause: The selected host cannot accommodate any additional users, or there was a communication failure between the personal computer and the host.

Action: Type y or n If you type y to retry the operation, you can select a different host to login to. If you prefer, you can attempt to make the selection that caused the error message. If the problem persists, consult the person who administers your system.

Selected Host Not in Table -- Try again

Cause: An invalid response, including a carriage return, was given to the Host selection menu for host file services.

Action: Give a valid host identification chosen from the Host menu.

The Access program hosts are not available

Cause: An emulation session cannot be started because no hosts are available. This could be a host problem or a problem in the communication link between the personal computer and the hosts.

Action: Retry the operation. If the problem persists, consult the person who administers your system.

The Access program software is not initialized

Cause: An attempt was made to invoke the emulator before the Access program was initialized. the Access program must be initialized before invoking the emulator.

Action: Initialize the Access program using the **pciinit** command.

There is not enough memory for a new session

Cause: The personal computer memory is inadequate.

Action: Retry the operation. If the message persists, install additional memory in your personal computer.

Unknown uexec error (*nn*)

Cause: An error occurred when **on** tried to initiate the requested AIX command on the host, but **on** was unable to determine the cause of the error. The internal error code is shown in *nn*.

Action: Confirm with the person who administers your network that the host computer and network services are available at this time.

PCI-vi Error Messages

Bad value for shiftwidth

Explanation: While using the Access program **vi** program, the command `set sw = 0` was typed. The value 0 is meaningless for **shiftwidth**.

User Response: Use a value other than 0 for **shiftwidth**.

Can't change hardtabs

Explanation: The value of **hardtabs** cannot be changed in the Access program **vi** editor.

User Response: None.

Can't exec shell

Explanation: While using the Access program **vi** program, this message appeared in response to the command **:sh** or **:cmd**. The **sh** command invokes the **command.com** shell by default. **command.com** is the DOS command interpreter, but the Access program **vi** shell variable can be set to invoke any program. To specify an alternative program, you have to include the program's full path name within the **COMSPEC** DOS environment variable. This program then becomes the default shell when the Access program **vi** is invoked. The shell variable can also be reset to refer to any program during an Access program **vi** session.

This error message is displayed when the shell cannot be executed. The most likely causes are insufficient memory to execute the shell or the shell program itself is missing or not fully specified.

User Response: Check whether the available memory is limited by suspended processes that still reside in memory. If there are suspended jobs consuming memory, finish or abort them to free additional memory. Also check that **command.com** (or whichever shell is set by the Access program **vi**) exists and is accessible.

Can't expand meta characters under DOS

Explanation: While using the Access program **vi** program, a command using a metacharacter was issued (for example, `:n *.c`, to edit files ending in .c). The Access program **vi**, under DOS, cannot expand metacharacters.

User Response: Spell out names in full rather than using metacharacters.

\escape not allowed in this context

Explanation: While using the Access program **vi** program, the backslash-escape combination was improperly used within a global command. For example, if you were to type `:g/string1/s//string2\`, the final backslash character is meaningless, and causes the operation to fail and the error message to appear. Such an error in standard AIX **vi** causes the editor to hang, but the Access program **vi** it does not.

User Response: Use the escape character properly.

Filtering not available under DOS

Explanation: With standard AIX **vi**, filters can be run on the contents of a file being edited. To run a filter with AIX **vi**, you specify the lines to be changed, then type an ! (exclamation mark) followed by the command name (for example, `:1,5!sort`). However, commands of this type cannot be executed from the Access program **vi** program.

User Response: None.

Improper ^V escape

Explanation: While using the Access program **vi** program, ^V was improperly used in a **map** command issued in **vi** mode. For example, the command `:map #1 xyz ^V^V` uses the ^V sequence in a meaningless way. Such an error in standard AIX **vi** causes the editor to hang, but the Access program **vi** it does not.

User Response: ^V must be used to quote a non-printing character.

Macros can't start with a digit

Explanation: While using the Access program **vi** program, an attempt was made to use the **map** command to map a string beginning with a digit to another string. Because digits are frequently used for line counts that apply to the Access program **vi** operations, using them to start a macrostring is likely to lead to errors. Macros, therefore, are not allowed to start with a digit.

User Response: When using the **map** command to map one string to another, do not start the first string with a digit.

Appendix C. Using a Dial-Up Modem

You can use the Access program with a dial-up modem connection to an AIX host in the same way that you use a direct RS-232 connection.

Since there are many kinds of dial-up modems and the exact steps they require for establishing communications vary, it exceeds the scope of this appendix to document the procedure in detail. However, in general, the steps are similar. Once the Access program initializes the communications port, you make a telephone connection to your AIX host. If you are using an automatic-dialing modem, you instruct the modem to dial the number from the keyboard. If you are using a manual modem, you dial the number and transfer control of the line to the modem after receiving a data tone.

This appendix describes these general steps using the internal Hayes Smartmodem for illustration. You should refer to the product manual for your dial-up modem for details on its operation.

When you plan to use only a dial-up modem, you should install the Access program for RS-232 only, following the installation instructions in Appendix A, "Installing AIX Access for DOS Users."

Before beginning your Access program session, be sure your modem is attached to, or installed correctly in, your personal computer and that it is connected to a standard telephone jack.

Establishing a Host File Services Session

The following procedure illustrates logging in to a host file services session over a modem:

1. Start your personal computer using your Access program disk.
2. Type the **LOGIN** command.
3. Choose the COM*n* Port connection path where *n* is the number of the port you want to use.
4. Choose the baud rate appropriate for your modem. The internal Hayes Smartmodem uses 1200 baud.
5. When the five-line login window appears, it remains blank. Establish the telephone connection to your AIX host.

   With the Hayes Smartmodem, type:

   AT DT *number*

   where *number* is the telephone number of your AIX host's dial-in dial-in modem.

   You hear the number being dialed, the phone ringing, and the host answer through the speaker in your personal computer. When the modem detects the host data tone, the words CONNECTED are displayed on your screen.
6. Press **Enter**. The AIX host identification and login banner should appear.
7. Continue with the RS-232 login sequence described in Chapter 4, "Using Your Computer as an AIX Terminal."

Establishing a Terminal Emulation Session

The following procedure illustrates logging in to a terminal emulation session over a modem:

1. Start your personal computer using your Access program disk.
2. Type the **EM** command.
3. Choose the COM*n* connection path where *n* is the number of the port you want to use.
4. On the Change Parameters menu, choose the communication parameters and baud rate appropriate for your modem. The internal Hayes Smartmodem uses 1200 baud and the default values for the remaining parameters.
5. When the terminal emulation screen appears, it remains blank. Establish the telephone connection to your AIX host.

   With the Hayes Smartmodem, type:

   AT DT *number*

   where *number* is the telephone number of your AIX host's dial-in modem.

   You hear the number being dialed, the phone ringing, and the host answer through the speaker in your personal computer. When the modem detects the host data tone, the words CONNECTED are displayed on your screen.
6. Press **Enter**. The AIX host identification and login banner should appear.
7. Log in to the AIX host and continue with your terminal emulation session as described in Chapter 4, "Using Your Computer as an AIX Terminal."

Appendix D. Extended Library - PCILIB

CONTENTS

About This Appendix D-3
What Is PCILIB? D-4
  Library Formats D-4
  Memory Models Supported D-4
  Include Files D-5
  Manifest Constants D-5
Data Structures D-6
    Message Queue Identifier D-6
    Message Operation Permissions D-6
    Semaphore Identifier D-7
    Semaphore Operation Permissions D-7
dflthost D-9
getuattr D-10
isvirtual D-11
mapd2u D-12
mapu2d D-13
msgctl D-14
msgget D-16
msgop D-18
semctl D-21
semget D-23
semop D-25
uchmod D-27
uexec D-28
ukill D-30
uren D-32
uwait D-33
vdrive D-34
vhost D-35

About This Appendix

PCILIB is a programming library that allows DOS programs written in C or assembly language to access the extended I/O control functions of the Access program **bridge** (**bridge.sys**).

The functions provided by the bridge can be very useful for implementing sophisticated DOS applications on top of the the Access program virtual file service. Such applications include office automation systems, integrated DOS and AIX mail handlers, and direct communication systems operating between distributed DOS and AIX applications.

PCILIB provides the DOS programmer with functions to do the following:

- Determine and change the AIX attributes of files on virtual drives.
- Find the virtual drive associated with a given DOS path.
- Map DOS file names to AIX file names.
- Map AIX file names to DOS file names.
- Find the drive numbers of virtual drives.
- Find the names of connected hosts.
- Execute AIX processes remotely from the DOS environment.
- Provide interprocess communications between personal computers and AIX hosts using message queues and semaphores.

This appendix assumes that you are familiar with DOS, the Lattice or Microsoft C programming environment, and AIX Access for DOS Users. Users who need more information on one of these topics should refer to the following documents:

- *AIX Access for DOS Users Administrator's Guide.* This document provides details on how to install the Access program.
- *Microsoft C Compiler User's Guide.* This document describes the C programming environment offered by the Microsoft Corporation.
- *Lattice C Compiler User's Guide.* This document describes the C programming environment offered by the Lattice Corporation.
- *AIX Operating System Technical Reference.* This document describes implementing programs using interprocess communications in AIX.

What Is PCILIB?

PCILIB is a library of DOS object modules, in Microsoft and Lattice C format, that can be linked with user-written DOS programs to take advantage of the advanced features of the the Access program virtual file service, as well as DOS and AIX interprocess communications facilities.

The following table summarizes the extended functions provided by PCILIB:

dflthost	Gets and sets the AIX host to be used for subsequent IPC operations.
getuattr	Gets the attributes of an AIX file on a virtual drive.
isvirtual	Returns the virtual drive number of a specified path.
mapd2u	Maps a DOS path name to an AIX path name.
mapu2d	Maps an AIX path name to a DOS path name.
msgctl	Provides message control operations.
msgget	Gets a message queue identifier.
msgrcv	Gets a message from a message queue.
msgsnd	Sends a message to a message queue.
semctl	Controls semaphore operations.
semget	Gets a set of semaphores.
semop	Performs semaphore operations.
uchmod	Changes the attributes of an AIX file on a virtual drive.
uexec	Executes an AIX program on a host.
ukill	Sends a signal to an AIX process.
uren	Renames an AIX file.
uwait	Polls the status of an AIX task initiated with **uexec**.
vdrive	Gets the virtual drive number of a virtual drive.
vhost	Gets the name of a connected AIX host.

Library Formats

The PCILIB diskettes contain a set of Lattice library modules and a set of Microsoft library modules.

Memory Models Supported

PCILIB supports all memory models for both the Lattice and Microsoft C compilers.

The following list identifies the PCILIB files:

pcilcd.lib	Small code, large memory model for Lattice C
pcilcl.lib	Large memory model for Lattice C
pcilcp.lib	Large code, small data model for Lattice C
pcilcs.lib	Small memory model for Lattice C.

pcimscc.lib Compact memory model for Microsoft C

pcimscs.lib Small memory model for Microsoft C

pcimscm.lib Medium memory model for Microsoft C

pcimscl.lib Large memory model for Microsoft C

The default memory models on the Microsoft and Lattice C compilers are the small memory models. Be sure to link the application program with the correct memory model library to ensure proper operation.

Include Files

The AIX Access for DOS Users distribution diskettes also contain three ***include*** files for C programs that make calls to the PCILIB functions. The include files are identified in the following list:

pcilib.h Defines the external PCILIB function call return types. This file must always be included in programs that use PCILIB function calls.

ipc.h Defines data structures that are used in IPC message queues and semaphore operations. This file must be included in programs making IPC calls.

memmdl.h Contains the manifest constants for compiling with either the Microsoft or Lattice C compilers. This file must always be included in programs that make PCILIB-supported calls.

Some of the PCILIB functions set the **errno** variable when they return an error condition. In order to examine the DOS **errno** variable, the calling program should include the AIX system file **errno.h**. Consult your C compiler user's guide to determine where to find this file and how to make use of the variable.

Manifest Constants

You must use one of two manifest constants (**LATTICE** or **MICROSOFT**) when you compile C programs that make PCILIB calls. You should use the **-d** command line option on the compiler to declare which constant to select from the file **memmdl.h**.

Data Structures

Purpose

Introduction to message queues and semaphores.

Description

This section describes message queues, semaphore permissions, and data structures.

Message Queue Identifier

A message queue identifier (*msqid*) is a unique positive integer created by a **msgget** system call. Each *msqid* has a message queue and a data structure associated with it. The data structure is referred to as **msqid_ds** and contains the following members:

```
struct ipc_perm msg_perm;       /* operation permission struct */
time_t msg_stime;               /* last msgsnd time */
time_t msg_rtime;               /* last msgrcv time */
time_t msg_ctime;               /* last change time */
                                /* Times measured in secs since */
                                /* 00:00:00 GMT, Jan. 1, 1970 */
ushort msg_cbytes;              /* current number of bytes on q */
ushort msg_qnum;                /* number of msgs on q */
ushort msg_qbytes;              /* max number of bytes on q */
ushort msg_lspid;               /* pid of last msgsnd operation */
ushort msg_lrpid;               /* pid of last msgrcv operation */
```

The *msg_perm* **ipc_perm** structure specifies the message operation permission (see "Message Operation Permissions"). This structure includes the following members:

```
ushort uid;                     /* user ID */
ushort gid;                     /* group ID */
ushort cuid;                    /* creator user ID */
ushort cgid;                    /* creator group ID */
ushort mode;                    /* r/w permission */
ushort seq;                     /* slot usage sequence number */
key_t  key;                     /* key */
```

The time of the last **msgsnd** operation is *msg_stime*. The time of the last **msgrcv** operation is *msg_rtime*. The time of the last **msgctl** operation that changed a member of the above structure is *msg_ctime*. The number of bytes currently on the queue is *msg_cbytes*. The *msg_qnum* member is the number of messages currently on the queue. The *msg_qbytes* member is the maximum number of bytes allowed on the queue. The process ID of the last process that performed a **msgsnd** operation is *msg_lspid*. The process ID of the last process that performed a **msgrcv** operation is *msg_lrpid*.

Message Operation Permissions

In the **msgop** and **msgctl** system call descriptions, the permission required for an operation is given as follows:

PERMISSION	MODE (octal)
Read by user	00400
Write by user	00200
Read, write by group	00060
Read, write by others	00006

Read and write permissions on an *msqid* are granted to a process if one or more of the following are true:

- The effective user ID of the process is superuser.
- The effective user ID of the process matches *msg_perm.[c]uid* in the data structure associated with *msqid*, and the appropriate bit of the User portion (0600) of *msg_perm.mode* is set.
- The effective user ID of the process does not match *msg_perm.[c]uid*, and the effective group ID of the process matches *msg_perm.[c]gid*, and the appropriate bit of the Group portion (060) of *msg_perm.mode* is set.
- The effective user ID of the process does not match *msg_perm.[c]uid*, and the effective group ID of the process does not match *msg_perm.[c]gid*, and the appropriate bit of the Other portion (06) of *msg_perm.mode* is set.

Otherwise, the corresponding permissions are denied.

Semaphore Identifier

A semaphore identifier (*semid*) is a unique positive integer created by a **semget** system call. Each *semid* has a set of semaphores and a data structure associated with it. The data structure is referred to as **semid_ds** and contains the following members:

```
struct ipc_perm sem_perm;        /* operation permission struct */
time_t sem_otime;                /* last operation time */
time_t sem_ctime;                /* last change time */
                                 /* Times measured in secs since */
                                 /* 00:00:00 GMT, Jan. 1, 1970 */
ushort sem_nsems;                /* number of sems in set */
```

The *sem_perm* **ipc_perm** structure specifies the semaphore operation permission (see "Semaphore Operation Permissions").

The *sem_nsems* member is equal to the number of semaphores in the set. Each semaphore in the set is referenced by a positive integer referred to as a *sem_num*. The *sem_num* values run sequentially from 0 to the value o f *sem_nsems*-1. The *sem_otime* member is the time of the last **semop** operation. The *sem_ctime* member is the time of the last **semctl** operation that changed a member of the above structure.

A semaphore is a data structure that contains the following members:

```
ushort semval;                   /* semaphore value */
short  sempid;                   /* pid of last operation  */
ushort semncnt;                  /* # awaiting semval > cval */
ushort semzcnt;                  /* # awaiting semval = 0 */
```

The *semval* member is a nonnegative integer. The *sempid* member is equal to the process ID of the last process that performed a semaphore operation on this semaphore. The *semncnt* member is a count of the number of processes that are currently suspended awaiting this semaphore's *semval* to become greater than its current value. The *semzcnt* member is a count of the number of processes that are currently suspended awaiting this semaphore's *semval* to become zero.

Semaphore Operation Permissions

In the **semop** and **semctl** system call descriptions, the permission required for an operation is given as as follows:

```
PERMISSION              MODE (octal)
 Read by user            00400
 Write by user           00200
 Read, write by group    00060
 Read, write by others   00006
```

Read and write permissions on a **semid** are granted to a process if one or more of the following are true:

- The effective user ID of the process is superuser.
- The effective user ID of the process matches *sem_perm.[c]uid* in the data structure associated with *semid*, and the appropriate bit of the User portion (0600) of *sem_perm.mode* is set.
- The effective user ID of the process does not match *sem_perm.[c]uid*, and the effective group ID of the process matches *sem_perm.[c]gid*, and the appropriate bit of the Group portion (060) of *sem_perm.mode* is set.
- The effective user ID of the process does not match *sem_perm.[c]uid*, and the effective group ID of the process does not match *sem_perm.[c]gid*, and the appropriate bit of the Other portion (06) of *sem_perm.mode* is set.

Otherwise, the corresponding permissions are denied.

Related Information

For more information, refer to "dflthost" on page D-9, "msgctl" on page D-14, "msgget" on page D-16, "msgop" on page D-18, "semctl" on page D-21, "semget" on page D-23, and "semop" on page D-25.

dflthost

Purpose

Get or set the current host for PCILIB IPC functions.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int dflthost( drvnum )
int drvnum;
```

Description

The **dflthost** subroutine allows the user to read and/or set which host is used for performing PCILIB IPC operations. A call to **dflthost** selects the host corresponding to the virtual drive indicated by *drvnum* (A = 1, B = 2, and so forth). This virtual drive may or may not be logged into a host. Initially, the host corresponding to the first virtual drive is selected.

If *drvnum* is 0, or any value that does not correspond to a virtual drive, then the current default host is not changed.

If a user program changes the default host, it should restore it to its original value before exiting.

Return Codes

The **dflthost** subroutine always returns what the current default host was before the call was made. The return value is an integer assigned as follows: A = 1, B = 2, and so forth.

Related Information

For more information, refer to "msgctl" on page D-14, "msgget" on page D-16, "msgop" on page D-18, "semctl" on page D-21, "semget" on page D-23, and "semop" on page D-25.

getuattr

Purpose

Gets AIX file attributes.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int getuattr( upath )
char *upath;
```

Description

The **getuattr** subroutine is used to determine the attributes of the specified AIX file.

The *upath* parameter is the AIX file specification for which the attributes are being requested. The *upath* parameter may include a virtual drive designation (for example, `D:`) indicating which host the file is stored on.

Return Codes

If the call succeeds, the AIX file mode is returned. If the call fails for any reason, a value of -1 is returned.

Related Information

For more information, refer to "isvirtual" on page D-11, and "uchmod" on page D-27.

isvirtual

Purpose

Returns the virtual drive number of a specified path.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int isvirtual( dpath )
char *dpath;
```

Description

The **isvirtual** subroutine returns the virtual drive number of the path specified by *dpath*.

Note: If the drive letter is included in *dpath*, the rest of the path is ignored.

Return Codes

If the specified DOS path is found on a virtual drive, the drive number of the device is returned. Drive A returns a 1, drive B returns a 2, and so forth.

If the specified path is not on a virtual drive, a value of 0 is return.

Related Information

For more information, refer to "vdrive" on page D-34.

mapd2u

Purpose

Maps a DOS path name to an AIX path name.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int mapd2u( upath, dpath )
char *upath;
char *dpath;
```

Description

The **mapd2u** subroutine returns a DOS path name in terms of AIX path name conventions (including the virtual drive designation).

The *upath* parameter is the AIX path name returned by the function. It is a fully qualified, null-terminated string with a drive designation that exactly corresponds to the DOS path.

The *dpath* parameter is the DOS path name. It must be provided as a null-terminated string.

Return Codes

On successful completion, a value of 0 is returned. If the translation fails (the DOS path name does not exist), a value of -1 is returned and the DOS error code is returned in **errno**.

Notes

No error is returned if a path component does not exist. The nonexistent component and remainder of the path are simply forced to lowercase. The calling program should ensure that the path exists before calling this function.

Related Information

For more information, refer to "isvirtual" on page D-11, and "mapu2d" on page D-13.

mapu2d

Purpose

Maps an AIX path name to a DOS path name.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int mapu2d( dpath, upath )
char *dpath;
char *upath;
```

Description

The **mapu2d** subroutine returns an AIX path name in terms of DOS path name conventions (including the virtual drive designation).

The *dpath* parameter is the DOS path name returned by the function. It is a fully qualified, null-terminated string that exactly corresponds to the AIX path.

The *upath* parameter is the AIX path name. It must be provided as a null-terminated string. The path name may include a drive designation to indicate which host the translation is performed on. The *upath* parameter uses normal AIX conventions (for example, forward slashes as directory separators).

Return Codes

On successful completion, a value of 0 is returned.

If the translation fails, a value of -1 is returned. The following are possible causes for failure; the AIX path name does not exist, access permission is denied, or the path contains an invalid directory component.

Notes

No error is returned if a path component does not exist. The nonexistent component and remainder of the path is simply forced to uppercase. The calling program should ensure that the path exists before calling this function.

Related Information

For more information, refer to "isvirtual" on page D-11, and "mapd2u" on page D-12.

msgctl

Purpose

Provides message control operations.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int msgctl( msqid, cmd, buf )
int msqid, cmd;
struct msqid_ds *buf;
```

Description

The **msgctl** subroutine provides a variety of message control operations as specified by *cmd*. The operations are performed on the current host (as specified in the most recent **dflthost** call). The following commands (*cmd*) are available:

IPC_STAT Place the current value of each member of the data structure associated with *msqid* into the structure pointed to by *buf*. The contents of this structure are defined in "Data Structures" on page D-6. The current process must have read permission to perform this operation.

IPC_SET Set the value of the following members of the data structure associated with *msqid* to the corresponding value found in the structure pointed to by *buf*:

```
msg_perm.uid
msg_perm.gid
msg_perm.mode                    /* only low 9 bits */
msg_qbytes
```

This command can only be executed by a process with an effective user ID of superuser or the value of *msg_perm.[c]uid* in the data structure associated with *msqid*. Only the superuser can raise the value of *msg_qbytes*.

IPC_RMID Remove the message queue identifier specified by *msqid* from the AIX system and destroy the message queue and data structure associated with it. This command can only be executed by a process that has an effective user ID of superuser or the value of *msg_perm.[c]uid* in the data structure associated with *msqid*.

Return Codes

On successful completion, a value of 0 is returned. If the call fails for any reason, a value of -1 is returned and **errno** is set to indicate the error.

The **msgctl** operation fails if one or more of the following are true:

EINVAL The *msqid* parameter is not a valid message queue identifier.

EINVAL The value of *cmd* is not valid.

EACCES The command specified by the *cmd* parameter is **IPC_STAT** and read permission is denied to the calling process (see "Data Structures" on page D-6).

EPERM The command specified by the *cmd* parameter is **IPC_RMID** or **IPC_SET**, and the effective user ID of the calling process is not superuser, and not equal to the value of *msg_perm.[c]uid* in the data structure associated with *msqid*.

EPERM The command specified by the *cmd* parameter is **IPC_SET**, and an attempt is being made to increase to the value of *msg_qbytes*, but the effective user ID of the calling process is not superuser.

EFAULT The *buf* parameter points to an invalid address.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "msgget" on page D-16, and "msgop" on page D-18.

msgget

Purpose

Gets a message queue identifier.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int msgget( key, msgflg )
key_t key;
int msgflg;
```

Description

The **msgget** subroutine returns the message queue identifier associated with *key* from the AIX host specified in the most recent **dflthost** call.

A message queue identifier with the associated message queue and data structure (see "Data Structures" on page D-6) are created for *key* if one of the following is true:

- The *key* parameter is equal to **IPC_PRIVATE**.
- The *key* parameter does not already have a message queue identifier associated with it, and *msgflg* is set to **IPC_CREAT**.

Upon creation, the data structure associated with the new message queue identifier is initialized as follows:

1. The *msg_perm.cuid*, *msg_perm.uid*, *msg_perm.cgid*, and *msg_perm.gid* members are set to the effective user ID and effective group IDs of the calling process.
2. The low-order nine bits of *msg_perm.mode* are set equal to the low-order nine bits of *msgflg*.
3. The *msg_qnum*, *msg_lspid*, *msg_lrpid*, *msg_stime*, and *msg_rtime* members are set to 0.
4. The *msg_ctime* member is set to the current AIX time.
5. The *msg_qbytes* member is set to the AIX system limit.

Return Codes

On successful completion, a message queue identifier is returned. Otherwise, a value of -1 is return and **errno** is set to indicate the error.

The **msgget** subroutine fails if one or more of the following are true:

EACCES A message queue identifier exists for *key*, but operation permission (see "Data Structures" on page D-6) as specified by the low-order nine bits of *msgflg* is not granted.

ENOENT A message queue identifier does not exist for *key*, and *msgflg* has the bits defined by **IPC_CREAT** cleared.

ENOSPC A message queue identifier is to be created, but the number of message queue identifiers exceeds the system maximum allowed for that host.

EEXIST A message queue identifier exists for *key*, but *msgflg* has the bits defined by **IPC_CREAT** and **IPC_EXCL** set.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "msgctl" on page D-14, and "msgop" on page D-18.

msgop

Purpose

Message operations.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int msgsnd( msqid, msgp, msgsz, msgflg)
int msqid;
struct msgbuf *msgp;
int msgsz, msgflg;

int msgrcv( msqid, msgp, msgsz, msgtyp msgflg)
int msqid;
struct msgbuf *msgp;
int msgsz;
long msgtyp;
int msgflg;
```

Description

The **msgop** functions perform operations on interprocess communications message queues. These operations are performed by the **msgsnd** and **msgrcv** subroutines, which send and receive the messages, respectively.

Sending Messages

The **msgsnd** subroutine sends a message to the queue associated with the message queue identifier specified by *msqid*. The current process must have write permission to perform this operation. The operations are performed on the current host (as specified in the most recent **dflthost** call). The *msgp* parameter points to a structure containing the message. This structure is composed of the following members:

```
long   mtype;                  /* message type */
char   mtext[];                /* message text */
```

The *mtype* member is a positive integer that can be used by the receiving process for message selection (see "Receiving Messages"). The *mtext* array is any text of length *msgsz* bytes. The *msgsz* parameter ranges from 0 to a system-imposed maximum.

The *msgflg* parameter specifies the action to be taken if either or both of the following are true:

- The number of bytes already on the queue is equal to *msg_qbytes*.
- The total number of messages on all queues systemwide is equal to the AIX system-imposed limit.

If one of these two events occurs and *msgflg* is *true*, the message is not sent and the DOS function returns immediately. If *msgflg* is *false*, the DOS function returns -1 and **errno** is set to **EIDRM**. Under no circumstances does the PCILIB **msgsnd** function allow the DOS task to become suspended.

The **msgsnd** operation fails and no message is sent if one or more of the following are true:

EINVAL The *msqid* parameter does not specify a valid message queue identifier.

EACCES Operation permission is denied to the calling process (see "Data Structures" on page D-6).

EINVAL The *mtype* member is less than 1.

EAGAIN The message cannot be sent for one of the reasons cited above, and *msgflg* is set to **C_NOWAIT**.

EINVAL The *msgsz* parameter is less than zero or greater than the system-imposed limit.

EFAULT The *msgp* parameter points to an illegal address.

On successful completion, the following actions are taken with respect to the data structure associated with *msqid* (see "Data Structures" on page D-6).

1. The *msg_qnum* member is incremented by 1.
2. The *msg_lspid* member is set to the process ID of the calling process.
3. The *msg_stime* member is set to the current time.

Receiving Messages

The **msgrcv** operation reads a message from the queue associated with the message queue identifier specified by *msqid* and places it in the structure pointed to by *msgp*. The current process must have read permission to perform this operation. This structure is composed of the following members:

```
long   mtype;                    /* message type */
char   mtext[];                  /* message text */
```

The *mtype* member is the type of the received message as specified by the sending process. The *mtext* array is the text of the message. The *msgsz* parameter specifies the size in bytes of *mtext*. The received message is truncated to *msgsz* bytes if it is larger than *msgsz* and *msgflg* has the bits defined by **MSG_NOERROR** set. The truncated part of the message is lost, and no indication of the truncation is given to the calling process.

The *msgtyp* parameter specifies the type of message requested, as follows:

- If *msgtyp* is equal to 0, the first message on the queue is received.
- If *msgtyp* is greater than 0, the first message of type *msgtyp* is received.
- If *msgtyp* is less than 0, the first message of the lowest type that is less than or equal to the absolute value of *msgtyp* is received.

The *msgflg* parameter specifies the action to be taken if a message of the desired type is not on the queue, as follows:

- If *msgflg* has the bits defined by **IPC_NOWAIT** set, the calling process immediately returns a value of -1 and **errno** is set to **ENOMSG**.
- If *msgflg* has the bits defined by **IPC_NOWAIT** cleared, the calling process returns a value of -1 and **errno** is set to **EIDRM**.

The **msgrcv** operation fails and no message is received if one or more of the following are true:

EINVAL The *msqid* parameter is not a valid message queue identifier.

EACCES Operation permission is denied to the calling process.

EINVAL The *msgsz* parameter is less than 0.

E2BIG The length of the *mtext* string is greater than *msgsz* and *msgflg* is not set to **MSG_NOERROR**.

ENOMSG The queue does not contain a message of the desired type and *msgtyp* is set to **IPC_NOWAIT**.

EFAULT The *msgp* parameter points to an invalid address.

On successful completion, the following actions are taken with respect to the data structure associated with *msqid* (see "Data Structures" on page D-6).

1. The *msg_qnum* member is decremented by 1.
2. The *msg_lrpid* member is set to the process ID of the calling process.
3. The *msg_rtime* member is set to the current time.

Return Codes

On successful completion, **msgsnd** returns a value of 0 and **msgrcv** returns the number of bytes actually placed into *mtext*.

If **msgsnd** or **msgrcv** return due to removal of *msqid* from the system, a value of -1 is returned and **errno** is set to **EIDRM**.

Otherwise, a value of -1 is returned and **errno** is set to indicate the error.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "msgctl" on page D-14, and "msgget" on page D-16.

semctl

Purpose

Controls semaphore operations.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int semctl( semid, semnum, cmd, arg )
int semid, cmd;
int semnum;
union semun {
    int val;
    struct semid_ds *buf;
    ushort *array;
} arg;
```

Description

The **semctl** subroutine provides a variety of semaphore control operations as specified by *cmd*. The operations are performed on the current host (as specified in the most recent **dflthost** call).

The following operations (*cmd*) are executed with respect to the semaphore specified by *semid* and *semnum*:

GETVAL Return the value of *semval* (see "Data Structures" on page D-6). The current process must have read permission to perform this operation.

SETVAL Set the value of *semval* to *arg.val*. The current process must have write permission to perform this operation. When this command is successfully executed the *semadj* value corresponding to the specified semaphore in all AIX processes is cleared.

GETPID Return the value of *sempid*. The current process must have read permission to perform this operation.

GETNCNT Return the value of *semncnt*. The current process must have read permission to perform this operation.

GETZCNT Return the value of *semzcnt*. The current process must have read permission to perform this operation.

The following operations return, then set every *semval* in the set of semaphores.

GETALL Place *semvals* into the array pointed to by *arg.array*. The current process must have read permission to perform this operation.

SETALL Set *semvals* according to the array pointed to by *arg.array*. The current process must have write permission to perform this operation. When this command is successfully executed, the *semadj* values corresponding to each specified semaphore in all AIX processes are cleared.

The following operations are also available:

IPC_STAT Place the current value of each member of the data structure associated with *semid* into the structure pointed to by *arg.buf*. The contents of this structure are defined in "Data Structures" on page D-6. The current process must have read permission to perform this operation.

IPC_SET Set the value of the following members of the data structure associated with *semid* to the corresponding value found in the structure pointed to by *arg.buf*:

```
sem_perm.uid
sem_perm.gid
sem_perm.mode                    /* only low 9 bits */
```

This command can only be executed by a process that has an effective user ID of superuser or to the value of *sem_perm.[c]uid* in the data structure associated with *semid*.

IPC_RMID Remove the semaphore identifier specified by *semid* from the AIX system and destroy the set of semaphores and data structure associated with it. This command can only be executed by a process that has an effective user ID of superuser or to the value of *sem_perm.[c]uid* in the data structure associated with *semid*.

Return Codes

On successful completion, the value returned depends on the operation (*cmd*) as follows:

GETVAL is the value of semval.

GETPID is the value of sempid.

GETNCNT is the value of semncnt.

GETZCNT is the value of semzcnt.

All others have a value of 0.

If the operation fails, a value of -1 is returned and **errno** is set to indicate the error.

The **semctl** subroutine fails if one or more of the following are true:

EINVAL The *semid* parameter is not a valid semaphore identifier.

EINVAL The *semnum* parameter is less than zero or greater than *sem_nsems*.

EINVAL The *cmd* parameter is not a valid command.

EACCES Operation permission is denied to the calling process (see "Data Structures" on page D-6).

ERANGE The operation specified by *cmd* is **SETVAL** or **SETALL** and the value to which *semval* is set is greater than the AIX system-imposed maximum.

EPERM The *cmd* parameter is equal to **IPC_RMID** or **IPC_SET**, and the effective user ID of the calling process is not superuser and not equal to the value of *sem_perm.[c]uid* in the data structure associated with *semid*.

EFAULT The *arg.buf* union member points to an invalid address.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "semget" on page D-23, and "semop" on page D-25.

semget

Purpose

Gets a set of semaphores.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int semget( key, nsems, semflg )
key_t key;
int nsems, semflg;
```

Description

The **semget** subroutine returns the semaphore identifier associated with *key*. The operations are performed on the current host (as specified in the most recent **dflthost** call).

A semaphore identifier and associated data structure and set containing *nsems* semaphores (see "Data Structures" on page D-6) are created for *key* if one of the following is true:

- The *key* parameter is equal to **IPC_PRIVATE**.
- The *key* parameter does not already have a semaphore identifier associated with it, and *semflg* is set to **IPC_CREAT**.

On creation, the data structure associated with the new semaphore identifier is initialized as follows:

1. The *sem_perm.cuid*, *sem_perm.uid*, *sem_perm.cgid*, and *sem_perm.gid* members are set to the effective user ID and effective group ID of the calling process.
2. The low-order nine bits of *sem_perm.mode* are set equal to the low-order nine bits of *semflg*.
3. The *sem_nsems* member is set to the value of *nsems*.
4. The *sem_otime* member is set to 0 and *sem_ctime* is set to the current AIX time.

Return Codes

On successful completion, a semaphore identifier is returned. Otherwise, a value of -1 is returned and **errno** is set to indicate the error.

The **semget** subroutine fails if one or more of the following are true:

EINVAL The value of *nsems* is either less than or equal to zero or greater than the AIX system-imposed limit.

EACCES A semaphore identifier exists for *key*, but operation permission (see "Data Structures" on page D-6) as specified by the low-order nine bits of *semflg* is not granted.

EINVAL A semaphore identifier exists for *key*, but the number of semaphores in the set associated with it is less than *nsems*, and *nsems* is not equal to zero.

ENOENT A semaphore identifier does not exist for *key*, and *semflg* does not have the bits defined by **IPC_CREAT** set.

ENOSPC A semaphore identifier is to be created, but the number of semaphore identifiers exceeds the system maximum allowed for that host.

EEXIST A semaphore identifier exists for *key*, but *semflg* is set to **IPC_CREAT** and **IPC_EXCL**.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "semctl" on page D-21, and "semop" on page D-25.

semop

Purpose

Performs semaphore operations.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <ipc.h>

int semop( semid, sops, nsops )
int semid;
struct sembuf **sops;
int nsops;
```

Description

The **semop** function is used to perform an array of semaphore operations on the set of semaphores associated with the semaphore identifier specified by *semid*. The operations are performed on the current host (as specified in the most recent **dflthost** call). The *sops* parameter is a pointer to the array of semaphore-operation structures. The *nsops* parameter is the number of such structures in the array. Each structure includes the following members:

```
short  sem_num;                /* semaphore number */
short  sem_op;                 /* semaphore operation */
short  sem_flg;                /* operation flags */
```

Each semaphore operation specified by *sem_op* is performed on the corresponding semaphore specified by *semid* and *sem_num*. The *sem_op* member specifies one of three semaphore operations as follows:

1. If *sem_op* is a negative integer and the current process has write permission, one of the following occurs:
   - If *semval* (see "Data Structures" on page D-6) is greater than or equal to the absolute value of *sem_op*, the absolute value of *sem_op* is subtracted from *semval*. Also, if *sem_flg* is set to **SEM_UNDO**, the absolute value of *sem_op* is added to the calling process's *semadj* value for the specified semaphore. (For more information, refer to the **exit** system call in *AIX Operating System Technical Reference*.)
   - If *semval* is less than the absolute value of *sem_op* and *sem_flg* is set to **IPC_NOWAIT**, **semop** returns immediately.
   - If *semval* is less than the absolute value of *sem_op* and *sem_flg* is not set to **IPC_NOWAIT**, **semop** returns -1 and **errno** is set to **EIDRM**.
2. If *sem_op* is a positive integer, and the current process has write permission, the value of **sem_op** is added to *semval*, and if *sem_flg* is set to **SEM_UNDO**, the value of *sem_op* is subtracted from the calling process's *semadj* value for the specified semaphore.
3. If *sem_op* is zero and the current process has read permission, one of the following occurs:
   - If *semval* is zero, **semop** returns immediately.
   - If *semval* is not equal to zero and *sem_flg* is set to **IPC_NOWAIT**, the **semop** subroutine returns immediately.
   - If *semval* is not equal to zero and *sem_flg* is not set to **IPC_NOWAIT**, **semop** returns -1 and **errno** is set to **EIDRM**.

Return Codes

On successful completion, the value of *semval* (at the time of the call for the last operation in the array pointed to by *sops*) is returned.

If **semop** returns due to the removal of a *semid* from the system, a value of -1 is returned and **errno** is set to **EIDRM**.

Otherwise, a value of -1 is returned and **errno** is set to indicate the error.

The **semop** subroutine fails if one or more of the following are true for any of the semaphore operations specified by *sops*:

EINVAL The *semid* parameter is not a valid semaphore identifier.

EFBIG The *sem_num* member is less than zero or greater than or equal to the number of semaphores in the set associated with *semid*.

E2BIG The *nsops* parameter is greater than the AIX system-imposed maximum.

EACCES Operation permission is denied to the calling process (see "Data Structures" on page D-6).

EAGAIN The operation would result in suspension of the calling process, but *sem_flg* is set to **IPC_NOWAIT**.

ENOSPC The AIX system limit on the number of individual processes requesting a **SEM_UNDO** is exceeded.

EINVAL The number of individual semaphores for which the calling process requests a **SEM_UNDO** exceeds the AIX system limit.

ERANGE An operation would cause a *semval* to overflow the AIX system-imposed limit.

ERANGE An operation would cause a *semadj* value to overflow the AIX system-imposed limit.

EFAULT The *sops* parameter points to an invalid address.

On successful completion, the value of *semid* for each semaphore specified in the array pointed to by *sops* is set to the process ID of the DOS server.

Notes

Any requested **UNDO** operations occur on a particular host when you log out of that host, rather than when the DOS application exits.

Only non-blocking operations are allowed from DOS programs. If a condition arises that would cause the DOS process to block, the function returns an error.

Related Information

For more information, refer to "Data Structures" on page D-6, "dflthost" on page D-9, "semctl" on page D-21, and "semget" on page D-23.

uchmod

Purpose

Change AIX file attributes.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int uchmod( dpath, mode )
char *dpath;
int mode;
```

Description

The **uchmod** subroutine is used to change the file attributes, or mode, of an AIX file name.

The *mode* parameter specifies what the access permissions are to be set to. (For a description of the various bits in the *mode* parameter, refer to the **chmod** system call in *AIX Operating System Technical Reference*.)

The *dpath* parameter is a DOS-style path name for the AIX file for which the permission mode is to be changed. The path may include a drive designation to indicate which host the operation is to be performed on.

Return Codes

On successful completion, a value of 0 is returned. If the call fails for any reason, a value of -1 is returned and a DOS error code is returned in **errno**.

Related Information

For more information, refer to "getuattr" on page D-10, and "isvirtual" on page D-11.

uexec

Purpose

Executes an AIX command.

Format

```
#include <pcilib.h>
#include <memmdl.h>

long uexec( drvnum, ucmdname, argv, envp, savestatus, error_code )
char *ucmdname, *argv[], *envp[];
int drvnum, savestatus, *error_code;
```

Description

The **uexec** subroutine starts the execution of an AIX command on an AIX host and returns the process ID. The AIX standard output, standard input, and standard error are all directed to **/dev/null**.

The *drvnum* parameter is the drive number of the host on which the command is executed. Drive letters and numbers correspond in the following way: A = 1, B = 2, and so forth. If *drvnum* is 0, the program is executed on the host associated with the current drive.

The *ucmdname* parameter points to a path name that identifies the AIX system and the file to be executed. This can be a load module or a shell command file. If the name contains no slash (/), the directories **/bin** and **/usr/bin** are searched in order.

The *argv* parameter is an array of character pointers to the parameters for the command. By convention, *argv[0]* points to the command name. The *argv* parameter is terminated by a null pointer.

The *envp* parameter is an array of character pointers to null-terminated strings. These strings constitute the environment for the new process. The *envp* parameter is terminated by a null pointer.

Return Codes

The *savestatus* parameter is an integer taking the following values:

1 Save exit status on termination.

0 Do not save exit status on termination.

When the process exits, the exit status is optionally saved depending on the value of the *savestatus* flag. The DOS **uwait** function may be used to poll for the exit status of the process.

On successful completion, the process ID of the child process returns. If the call fails for any reason, a value of -1 is returned, and *error_code* is set to indicate the error. The return value is a 32-bit integer.

The error codes for this call are:

1 Error in request service (network error).

2 The specified *drvnum* is not connected to a host.

3 The AIX **exec** system call failed.

4 Invalid format.

5 DOS memory allocation error.

If the AIX file-server process exits due to logging out, any AIX processes started with **uexec** are stopped unless steps are taken to avoid this (refer to the **signal** system call in *AIX Operating System Technical Reference*).

When using shell features such as I/O redirection, use **sh** or **csh** as *ucmdname* and **-c** as the first parameter. The command line being executed by the shell should be the second argument.

Notes

The signal sent to processes executed by **uexec** when the server exits is **SIGTERM** instead of **SIGHUP**.

Related Information

For more information, refer to "ukill" on page D-30, and "uwait" on page D-33.

ukill

Purpose

Sends a signal to an AIX process or group of processes.

Format

```
#include <pcilib.h>
#include <memmdl.h>
#include <sys/types.h>
#include <signal.h>

int ukill( drvnum, pid, sig )
int drvnum;
pid_t pid;
int sig;
```

Description

The **ukill** subroutine sends a signal to an AIX process or group of processes. Specify the process or group of processes with the *pid* parameter. The **drvnum** parameter is the number of the virtual drive associated with the AIX host running the process or processes (drive A = 1, drive B = 2, and so forth). If *drvnum* is 0, the signal is sent to the host associated with the current drive.

Specify the signal to be sent with *sig*. The value of **sig** may be 0 or any valid AIX signal value (refer to **signal** in *AIX Operating System Technical Reference* for a list of valid signals). For compatibility with non-AIX systems, the value of *sig* may also be the negative of one of the valid AIX signals. If *sig* is 0, error checking is performed but no signal is actually sent. This can be used to check the validity of *pid*.

The real or effective user ID of the sending process must match the real or effective user ID of the receiving process (*pid*), unless the user ID of the sending process is superuser.

Several processes, including process 0 and 1, are special system-created processes. Of these, only process 1 (the initialization process) may ever be signaled. If *pid* is greater than 0 and *sig* is greater than 0, then the signal specified by *sig* will be sent to the process specified by *pid*. The value of *pid* may be 1, but it may not be the process ID of any other system-created process.

If *pid* is 0, the absolute value of *sig* will be sent to each process (except the system-created processes) with a process group ID matching the process group ID of the sending process. The sending process's group ID may not be 0.

If *pid* is -1 and the effective user ID of the sending process is not superuser, the absolute value of *sig* will be sent to each process with a real user ID that matches the effective user ID of the sending process.

If *pid* is -1 and the effective user ID of the sending process *is* superuser, the absolute value of *sig* will be sent to all process, except the system-created processes.

If *pid* is negative, but not -1, the absolute value of *sig* is sent to each process with a process group ID that matches the absolute value of *pid*.

If *pid* is positive and *sig* is negative, the absolute value of *sig* is sent to each process with a process group ID that matches *pid*.

Return Codes

On successful completion, a value of 0 is returned. If the call fails for any reason, a value of -1 is returned, and **errno** is set to indicate the error.

ukill fails and no signal is sent if one or more of the following are true:

EINVAL The value of *sig* is not a valid signal number.

EINVAL The *pid* parameter is the process ID of one of the system-created process which cannot be signaled.

ESRCH No process can be found that corresponds to the process specified by *pid*.

ESRCH The *pid* parameter is 0 and the process group ID of the sending process is 0.

EPERM The user ID of the sending process is not superuser, and its real or effective user ID does not match the real or effective user ID of the process specified by *pid*.

Related Information

For more information, refer to "uexec" on page D-28, and "uwait" on page D-33.

uren

Purpose

Renames an AIX file.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int uren( oldupath, newupath )
char *newupath;
char *oldupath;
```

Description

The **uren** subroutine is used to rename AIX files. The file to be renamed must exist, and the DOS user must have access permission to it.

The *oldupath* parameter is the AIX file to be renamed. The name may include a drive designation to select the host.

The *newupath* parameter is the name to be given to the file.

Return Codes

If any error occurs, a value of -1 is returned.

Related Information

For more information, refer to "isvirtual" on page D-11, and "mapd2u" on page D-12.

uwait

Purpose

Polls for the exit status of an AIX process.

Format

```
#include <pcilib.h>
#include <memmdl.h>

long uwait( drvnum, status, blockflag )
int  blockflag, drvnum, *status;
```

Description

The **uwait** subroutine checks to see if an AIX command started with the **uexec** function has stopped. The AIX program must have been executed with the option to save the exit status.

The *drvnum* parameter is the drive number of the host on which to check. Drive letters and numbers correspond in the following way: A = 1, B = 2, and so forth. If *drvnum* is 0, **uwait** is performed on the host associated with the current drive.

Return Codes

The low byte of status is the exit status supplied by the terminating process or, if it was terminated by a signal, the signal that terminated it. The following values are returned in the high byte of the status field:

0 The low byte is the exit status.

1 The low byte is the signal terminating the process.

2 The low byte is signal terminating process and generating a core dump.

3 The process was not started by **uexec**, or the status was already collected.

The *blockflag* parameter controls whether the call is blocking or non-blocking. If *flag* is 1, the call blocks. If *flag* is 0, the call does not block. Only the non-blocking form of this call is currently supported.

The return values for this call are defined as follows:

-1 There are no child processes for which *status* is being saved.

0 No child processes have exited.

>0 This is the exited child process ID, and *status* is set.

Note: The return value is a 32-bit integer.

Related Information

For more information, refer to "uexec" on page D-28, and "ukill" on page D-30.

vdrive

Purpose

Returns the virtual drive number of a virtual drive.

Format

```
#include <pcilib.h>
#include <memmdl.h>

int vdrive( vdrvnum )
int vdrvnum;
```

Description

The **vdrive** subroutine returns the drive numbers of virtual drives.

The *vdrvnum* parameter is an integer that specifies which drive number to return. If *vdrvnum* is 0, the number of virtual drives is returned. If *vdrvnum* is a value between 1 and the number of virtual drives, the drive number of the corresponding virtual drive is returned.

Return Codes

The drive numbers returned by **vdrive** are as follows: A = 1, B = 2, and so on. These drives may or may not be currently logged in.

If no virtual drives are associated with the personal computer, or *vdrvnum* is greater than the number of virtual drives, a value of 0 is returned.

Related Information

For more information, refer to "isvirtual" on page D-11, and "vhost" on page D-35.

vhost

Purpose

Returns the name of a currently connected host.

Format

```
#include <pcilib.h>
#include <memmdl.h>

char* vhost( drvnum )
int drvnum;
```

Description

The **vhost** subroutine returns the name of the host attached to the virtual drive number specified by *drvnum*. Drive letters and numbers correspond in the following way: A = 1, B = 2, and so forth.

Return Codes

If *drvnum* corresponds to a logged-in virtual drive, **vhost** returns the name of the host associated with the virtual drive specified by *drvnum*.

If *drvnum* is out of range, or if the drive that was specified is associated with a host not currently logged in, then **vhost** returns a null string.

The name returned is stored in a static buffer. Subsequent calls to **vhost** will overwrite this buffer.

Related Information

For more information, refer to "isvirtual" on page D-11, and "vdrive" on page D-34.

Index

A

Access program
  files 5-4
  installing A-3
  installing on diskette drive system A-10
  installing on fixed disk drive A-18, A-20
  installing on RAM disk A-14
  invoking vi editor 6-4
  messages B-1
  overview of installing A-7
  putting on a subdirectory 5-5
  putting on a virtual drive 5-6
  tailoring 5-3
AIX
  character flow control 4-12
  DOS file differences 2-10
  executing processes 2-17
  file permissions 2-14
  file protection 2-14
AIX Access for DOS Users
  description 1-4
AIX vi editor
  differences with Access program 6-6
aix2dos 7-7
aix2dos command 7-7
Alt key 4-12
asynchronous communication adapter
  installing A-4

B

backspace key 4-12
baud rate
  changing 4-7
  changing host 4-9
  default 4-7
  selecting 4-9

C

Caps Lock key 4-12
Change Parameters menu
  F6 key 4-14
  parameters 4-6
changing
  job table size 5-9
  maximum number of virtual drive 5-8
Close, Open session menus
  under terminal emulation 4-10
commands
  aix2dos 7-7
  Data Structures D-6
  dflthost D-9
  doswhat 7-9
  dos2aix 7-10
  em 7-11
  exrecovr 7-12
  getuattr D-10
  isvirtual D-11
  jobs 7-13
  kill 7-14
  login 7-16
  LOGOUT 4-10, 7-18
  mapd2u D-12
  mapu2d D-13
  msgctl D-14
  msgget D-16
  msgop D-18
  nty 7-19
  on 3-3, 7-20
  pciinit 7-24
  printer 4-12, 7-25
  semctl D-21
  semget D-23
  semop D-25
  setdebug 7-28
  uchmod D-27
  udir 7-32
  uexec D-28
  ukill D-30
  uren D-32
  uwait D-33
  vdrive D-34
  vhost D-35
  vi 7-33
communication parameters
  baud rate 4-7
communications parameters 4-5
  changing 4-5, 4-9, 4-14
  defaults 4-6
  displaying current 4-14
  parity 4-7
  tabs 4-7
  terminal mode 4-6
compatibility 1-5
configuration options
  vi editor 6-5
connection path
  changing communications parameters 4-5
  changing current path 4-10
  closing current path 4-10, 4-14
  current 4-11
  retaining current path 4-10
  RS-232 4-5
copy protection 1-5
creating files
  in host file services 2-9
Ctrl key 4-12

cursor keys 4-13

D

Data Structures D-6-D-8
Delete key 4-13
detached jobs
  stopping 3-12
detached processes
  keeping track of 3-10
  methods of 3-9
  reattaching to 3-11
  redirecting 3-14
dflthost D-9
dial-up modem
  using C-1
directory
  displaying AIX-style 2-12
  permissions 2-12
diskette drive system
  installing Access program A-10
  making backup diskettes A-8
diskettes
  making backups A-18
DOS
  AIX file differences 2-10
  applications 2-16
  changing 5-8
  commands 2-16
  expanding environment space 3-8
  files 5-8
  input 3-13
  maximum number 5-8
  output 3-13
  pipes 3-13
  putting applications on a virtual drive 5-6
doswhat 7-9
doswhat command 7-9
dos2aix 7-10
dos2aix command 7-10

E

em 7-11
em command 7-11
EM.SES file 4-7, 4-8
  session file 4-10
EMSES 4-7, 4-8
End key 4-13
environment considerations 7-21
environment variables
  ONNAME 5-10
  ONPREFIX 5-10
  setting 3-7
  setting terminal emulation 5-11
  TERM environment variable 4-8, 4-9
error
  standard 7-22
error messages B-1
  during Access program installation A-14, A-20
Esc key 4-12
Exiting terminal emulation 4-14
EXPORT 3-7
exrecovr 7-12
exrecovr command 7-12

F

file
  recovery for vi editor 6-6
file services
  switching between 4-9
  switching from emulation 4-9
  switching from terminal emulation 4-14
  switching to emulation 4-10
file sharing
  description 2-16
files
  Access program 5-4, 5-6
  accessing files 2-10
  changing 2-13, 5-8
  changing modes 2-15
  converting 2-14
  creating files 2-10
  display modes 2-15
  DOS and AIX differences 2-10
  ending services session 2-20
  establishing a host services session C-1
  maximum number 5-8
  open 5-8
  permissions 2-14, 2-16
  putting on a virtual drive 5-6
  receiving 4-15
  sending 4-15
  services session 2-4
  sharing 2-16
  system 2-9
  under DOS 5-8
  vi working 6-5
fixed disk
  installing Access program A-18, A-20
  making backups A-18
function keys 4-13
  Alt and F10 key 4-14
  Alt and F6 key 4-14
  Help menu 4-12
  selecting from Help menu 4-12
F1 key 4-13
F10 key 4-14
F2 key 4-13
F3 key 4-13
F4 key 4-13
F5 key 4-14
F6 key 4-14
F7 key 4-14
F8 key 4-14
F9 key 4-14

G

getuattr D-10

H

hardware requirements 1-4
Help menu 4-11
  displaying 4-14
Home key 4-13
host
  establishing a file services session C-1
  minimum requirements 1-4
host file service
  switching between 4-9
host file services
  definition 2-3

I

IBM Token-Ring Network PC Adapter
  installing A-5
initialization of Access program
  requirements for, 5-4
input
  standard 7-22
Insert key 4-13
installing
  Access program on diskette drive system A-10
  Access program on one diskette-drive system A-13
  Access program on two diskette-drive system A-8
  Access program overview A-7
  asynchronous communication adapter A-4
  IBM Token-Ring Network PC Adapter A-5
  network interface adapter A-4
  Ungermann-Bass Network Interface Adapter A-4
  3Com EtherLink adapter A-4
isvirtual D-11

J

job
  changing table size 5-9
  control 3-9, 7-23
  detached 7-23
  saving output from 3-12
  table 3-9
jobs 7-13
jobs command 7-13

K

key
  combinations 4-14
  PC Scancode 4-14
keyboard features
kill command 7-14-7-15

L

LAN
  logging 2-5
linking
  on command on the virtual drive 3-6
local DOS
  switching between 4-9
logging
  additional host 2-7
  command line 2-8
  connection path 4-7
  from host file services 2-20
  in to host file services 2-4
  LAN 2-5
  RS-232 2-6
logging in
  over RS-232 connection path 4-8
login command 7-16-7-17
LOGOUT command 7-18
  description 4-10
  from terminal emulation 4-10

M

mapd2u D-12
mapu2d D-13
messages B-1
minimum requirements
  host 1-4
  personal computer 1-4
modem
  using Access program with C-1
modes
  changing 2-15
  displaying 2-15
msgctl D-14-D-15
msgget D-16-D-17
msgop D-18-D-20

N

naming files
  differences between DOS and AIX 2-10
network interface adapter
  installing A-4
  3Com EtherLink A-4

network types
list of available A-11, A-20
selecting, for record locking and file sharing 2-16
nty command 7-19
Num Lock key 4-13
numeric keypad 4-13

O

on command 7-20, 7-23
breaking out of 3-8, 7-22
cautions 3-15
description 3-3
environment variables 5-10
linking on the virtual drive 3-6
renaming 3-5
restrictions 3-15
search path and environment considerations 7-21
setting variables 5-9
summary of restrictions and cautions 3-15
using 3-4
on utilities
description 3-3
one diskette-drive system
installing A-13
output
standard 7-22

P

Package Types A-6
page down key 4-13
parity
default 4-7
PATH 3-7
PC Scancode
key combinations 4-14
pciinit command 7-24
permission mode
definition of 2-14
permission modes
changing 2-15
displaying 2-15
effect on DOS application 2-16
effect with DOS commands 2-16
for directories 2-16
personal computer
minimum requirements 1-4
personal computer keyboard
break signal 4-13
description 4-12
Page Up key 4-13
RS-232 connection path 4-13
pipes 3-13, 7-22
detached processes 3-14
print
local 2-17
remote 2-17
print command 2-17
Print Screen key 4-12
printer 7-25
setting environment 5-9
PRINTER command 7-27
printing
and local drive 7-25
default settings 2-18
DOS command 7-25
from remote drive 7-25
in terminal emulation 4-12
local drive 2-17
on local printer 4-12
time-out periods 2-19
under DOS 2-17

R

RAM disk
installing Access program A-14
making backups A-13
receiving files 4-15
record locking
and record backups 2-16
description 2-16
selection of network types 2-16
redirecting 3-13, 7-22
detached processes 3-14
input 3-13
output 3-13
renaming files 2-13

S

screen saving 4-7, 4-8
screen scrolling
page down key 4-13
search paths 7-21
setting 5-7
setting default 5-8
semctl D-21-D-22
semget D-23-D-24
semop D-25-D-26
sending files
in terminal emulation sessions 4-15
services session
establishing a host file C-1
session file
scrolling through 4-13
setdebug Command 7-28-7-31
standard
error 7-22
input 7-22
output 7-22
status line
keyboard status 4-11
LED 4-11

session status 4-11
Subdirectory
Access program 5-5
supported terminal emulation
functions 4-3

T

Tab key 4-13
tabs
setting 4-7
TERM environment variable 4-8, 4-9
terminal capabilities
setting 4-8, 4-9
terminal emulation
beginning a session 4-4
Change Parameters menu 4-9
closing a session 4-10
determining size of 4-7, 4-8
EM.SES file 4-9, 4-10
ending a session 4-14
environment variables 5-11
over connection path 4-8
over RS-232 4-8
requirements for, 5-4
saving screens 4-7, 4-8, 4-10
screen saving 4-7, 4-8, 4-10
session files 4-7, 4-8
setting 5-11
setting environment 4-10
status line 4-15
switching between 4-9
switching from file services 4-10
switching to file services 4-9, 4-14
transferring files during 4-15
terminal emulation session
establishing C-2
terminal-locking
with print processes 4-12
terminal mode 4-6
text formats
AIX 2-13
DOS 2-13
transferring files
between AIX and personal computer 4-15
file transferring 4-15
Open file menu 4-14, 4-15
transfer speed options 4-16
two diskette-drive system
installing A-8
typewriter
standard area 4-12

U

uchmod D-27
udir command 7-32
uexec D-28-D-29
ukill D-30-D-31
UMASK 2-15
Ungermann-Bass Network Interface Adapter
installing A-4
UPATH 3-7
uren D-32
uwait D-33

V

vdrive D-34
vhost D-35
vi Command 7-33-7-34
vi editor
configuration options 6-5
differences with AIX 6-6
file recovery 6-6
invoking 6-4
working files 6-5
virtual drive
Access program 5-6
changing maximum number of virtual drive 5-8
definition of 2-9
DOS applications 5-6
DOS command limitations of 2-9
files 5-6
storing Access program files on 5-6

Numerics

3Com EtherLink adapter
installing A-4

Reader's Comment Form

IBM AIX Access for DOS Users:
User's Guide

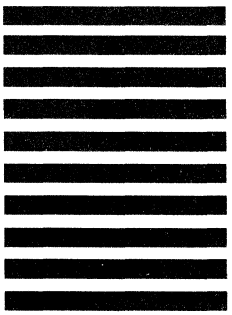
SC23-2041-0

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact your IBM representative, your authorized IBM dealer, or authorized IBM remarketer in your area.

Comments:

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department 997, Building 998
11400 Burnet Rd.
Austin, Texas 78758-3493

Fold

Fold

Cut or Fold Along Line

Fold and Tape

Please Do Not Staple

Fold and Tape

© IBM Corp. 1987, 1988
All rights reserved.

International Business
Machines Corporation
11400 Burnet Road
Austin, Texas 78758

Printed in the
United States of America

SC23-2041-0

SC23-2041-00